A Cluster-Based Framework for Contrast Preservation During Color Space Transformations

by

Cheryl Lau

M.S., Columbia University, 2006
B.S., Columbia University, 2004

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate Studies

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

March 2012

Abstract

Common tasks such as printing, displaying, and visualizing images may require a transformation of the input image from its source color space to a target color space. Such transformations include converting color images to grayscale for printing, mapping images to the gamuts of target display devices, preparing images for color deficient viewers, and fusing multispectral or multiprimary data into tristimulus images. Each of these transformations has a straightforward, standard mapping, but it often involves a loss of information due to dimensionality reduction or differing constraints for the source and target spaces. In the extreme case, contrast is completely lost when different colors in the source space map to the same color in the target space, an effect known as metamerism.

We present a framework for mapping an image from a source color space to a target color space in a way that preserves as much of the local contrast from the source image as possible while staying as faithful as possible to the standard mapping. We adopt a cluster-based approach in which the clusters represent local areas in the image, and the differences between the clusters represent local contrasts between neighboring areas. Our optimization translates clusters optimally to enhance local contrast without making the result seem unnatural. We apply our method to color to gray conversion, gamut mapping, image optimization for color deficient viewers, and conversion of multispectral and multiprimary images to tristimulus images. In each application, our method produces realistic, detail-preserving output images within their target spaces.

Preface

The work in this thesis is based on the following publication, which is joint work with Wolfgang Heidrich and Rafał Mantiuk. Dr. Heidrich had the initial, high-level idea. Dr. Mantiuk participated in discussions in the early stages of the project. The author designed and implemented the software, wrote the initial draft of the paper, which was subsequently edited by both Dr. Heidrich and the author, and created the project poster and video, which were presented at ICCV 2011.

Publications

C. Lau, W. Heidrich, R. Mantiuk. "Cluster-Based Color Space Optimizations". In *Proc. IEEE International Conference on Computer Vision*, pp. 1172–1179, 2011.

Table of Contents

List of Tables

List of Figures

Glossary

σ sigma for Gaussian used to calculate achromaticity weights.

E_L achromaticity term.

λ_L achromaticity term weight.

a_{ij} additional magnitude for edge (i, j).

$\overline{b}(\lambda)$ CIE 1931 2° color matching function for B primary.

M cluster's inverse covariance matrix.

μ cluster mean.

r cluster radius.

$\mathcal{V}$ set of vertices in graph, set of clusters.

$\mathbf{d}_i$ difference vector for cluster i.

$\mathbf{d}_q$ difference vector for pixel q.

ω_{qi} weight on cluster i's difference vector, for pixel q.

δ disk radius used in dilation.

$\mathcal{E}$ set of edges in graph.

(x, y) spatial coordinates.

(x', y') weighted spatial coordinates.

r_m subscript m indicates values for mapped clusters.

r_o subscript o indicates values for original clusters.

$C()$ function that converts colors from one color space to another color space.

$F()$ function that transforms vectors in target space.

$N()$ function that returns the absolute value of negative numbers or zero for non-negative numbers.

$P()$ projection operator, projects clusters to target space.

$S()$ function that clamps negative numbers and scales to [0,1].

$\overline{g}(\lambda)$ CIE 1931 2° color matching function for G primary.

$\mathcal{G}$ cluster graph.

H hue matrix from hue term.

E_H hue term.

λ_H hue term weight.

h_I image height (pixels).

w_I image width (pixels).

z iteration number.

l number of JNDs per length subtended by a 2° visual field.

L lightness matrix from achromaticity term.

Q matrix that projects onto achromatic axis.

W matrix containing achromaticity weights.

$\mathbf{m}$ vector of initially mapped cluster colors from target space.

m_{ij} distance between initially mapped clusters i and j.

$\mathbf{m}_{ij}$ vector between initially mapped colors for clusters i and j.

k maximum magnitude increase allowed.

k_p maximum magnitude increase allowed, expressed as percentage.

h_M display height (pixels).

ρ display's resolution (pixels/unit distance).

w_M display width (pixels).

N number of clusters.

Z number of iterations.

$\mathbf{x}$ vector of optimized cluster colors from linear least squares optimization.

$\mathbf{o}$ vector of original cluster colors from source space.

o_{ij} distance between original clusters i and j.

$\mathbf{o}_{ij}$ vector between original colors for clusters i and j.

υ amount of overlap for cluster pair (i, j).

g_i length of shortest path in graph from cluster i to nearest critical cluster.

g_{ij} length of shortest path in graph from edge (i, j) to nearest critical edge.

τ_i per cluster weight on cluster i.

τ_{ij} per edge weight on edge (i, j).

$V(\lambda)$ CIE 1924 2° photopic luminous efficiency curve.

$\mathbf{p}$ vector of projected output colors for clusters.

E_P previous iteration term.

λ_P previous iteration term weight.

$\overline{r}(\lambda)$ CIE 1931 2° color matching function for R primary.

E_M regularization term.

λ_M regularization term weight.

c sigmoid center.

κ sigmoid steepness.

ψ_{ij} sigmoidal weight.

n number of source dimensions.

$\mathcal{S}$ set of colors in source space.

$\mathbf{s}$ vector in source space.

b number of units in the source space equivalent to one JND.

$\mathcal{U}$ set of points in spatio-chromatic space.

$\mathbf{u}$ vector in spatio-chromatic space.

m number of target dimensions.

$\mathcal{T}$ set of colors in target space.

E_T target term.

$\mathbf{t}_{ij}$ target vector for edge (i, j).

v viewing distance from display.

$\overline{x}(\lambda)$ CIE 1931 2° color matching function for X primary.

$\overline{y}(\lambda)$ CIE 1931 2° color matching function for Y primary.

$\overline{z}(\lambda)$ CIE 1931 2° color matching function for Z primary.

HDR high dynamic range.

ITU-R BT.601 standard for standard-definition television published by the International Telecommunication Union Radiocommunication Sector (ITU-R), abbreviated as Rec. 601 or BT.601 or CCIR 601 [Recoa].

ITU-R BT.709 standard for high-definition television published by the International Telecommunication Union Radiocommunication Sector (ITU-R), abbreviated as Rec. 709 or BT.709 [Recob].

JND just noticeable difference.

LDR low dynamic range.

Rec. 601 abbreviation for the standard ITU-R BT.601 [Recoa].

Rec. 709 abbreviation for the standard ITU-R BT.709 [Recob].

Acknowledgements

I would like to acknowledge my supervisor, my committee, my family, and my friends for their support during my doctoral degree. I thank my supervisor Wolfgang Heidrich for the opportunity to do research and for his support and guidance throughout my five and half years at UBC. I thank my supervisory committee members Bob Woodham and Jim Little for the time and effort they invested in providing me with feedback during my doctoral program. I thank my examination committee, which includes the aforementioned supervisory committee, university examiners Michiel van de Panne and Panos Nasiopoulos, external examiner Bruce Gooch, and examination chair Derek Gates, for taking the time to read my thesis and serve on this committee. I thank my colleagues, my lab, both PSM and Imager, past and present, for numerous discussions, laughs, and comedic situations. I thank my family and friends for their love and support.

Chapter 1

Introduction

Color and multispectral images, as reproductions of real world scenes, often contain more information than can be reproduced on some printing and display devices. In this case, a reduction in the number of color channels, gamut, or number of spectral bands is necessary for printing, display, and visualization purposes. For example, to print a color image on a grayscale printer, the color image must be reduced to a single grayscale dimension, and chromatic contrasts can be lost. When displaying an image on a device with a smaller gamut, the device might not be capable of accurately reproducing some of the image's original colors. Similarly, when displaying an image to a color deficient viewer, the viewer will not perceive all of the original image colors as a person with normal vision would. Visualizing multispectral images on a conventional display requires reducing the many spectral bands to a tristimulus image, thus eliminating information during the conversion. For each of these situations, there exists a straightforward, standard mapping to the reduced space, but the transformation inherently incurs a loss of information as shown in Figures 1.1 and 1.2. The commonality among these problems inspires the central idea of this thesis:

> *We present a framework for mapping an image from a source color space to a target color space in a way that preserves information from the source image while staying faithful to the standard mapping. Our unified framework is a cluster-based approach which we apply to a variety of color space transformations including color to gray conversion, color gamut mapping, image optimization for color deficient viewers, and multispectral image fusion.*

With the transformations in our framework, there are two sources of information loss. The first is a loss of dimensionality, and the second is due to a difference in volume constraints within the same dimensional space. In color to grayscale conversion, the chromatic contrast loss occurs through the dimensionality reduction from three dimensions to one dimension. Similarly, some color deficient viewers only perceive two of the three color dimen-

sions, and multispectral image fusion also suffers from loss of dimensionality. Gamut mapping, on the other hand, is affected by the second cause of information loss. In this case, the source and target spaces have the same dimensionality but represent different volumes in that multidimensional space. Regardless of the cause of information loss, each can lead to metamerism, a phenomenon in which different source colors map to the same color in the target space. With metamerism, contrast between the colors is lost completely. Our method is devised to deal with transformation problems that suffer from a loss of contrast due to these two sources.

The goals of our framework are to preserve local contrasts from the source space within constraints of the target space and to maintain naturalness by staying close to a standard projection. Within the context of color to gray conversion, our goal is to preserve chromatic contrasts from the original color image as grayscale contrasts in our output gray image. Meanwhile, the competing goal is to stay close to the standard grayscale projection so that our result is a plausible grayscale version of the original image. For gamut mapping, our goal is to preserve out-of-gamut details while remaining close to the standard clipping result to maintain color accuracy for the in-gamut colors. To optimize images for color deficient viewers, our method aims to capture the color contrasts visible to humans with normal vision within the space of colors visible to color deficient viewers without deviating too far from how the color deficient viewer sees the original image. In multispectral image fusion, our goal is to preserve the differences between spectral metamers while staying close to how the human visual system naturally sees the scene for the purpose of creating a plausibly realistic result. These applications all share common goals: preserve contrasts lost during the standard mapping without deviating too far from it.

With these goals in mind, we adopt a cluster-based method for dealing with both types of information loss in a unified manner. Our method starts with a standard projection from the source space to the target space. This standard mapping defines a target appearance for our optimization. We then adopt a semi-local approach that operates on clusters. First, we group pixels into clusters according to both their spatial relationship and their similarity in the source color space. We then define a graph that represents local contrasts between clusters, especially those contrasts between metameric regions. Then, we find optimal cluster translations that restore lost contrast while maintaining proximity to the original colors. Finally, we transfer the cluster movements back to the pixels during a blending step.

Since our clustering is based on both spatial and chromatic spacing, we have in effect chosen a semi-local approach over a pure global or pure lo-

cal solution. Local contrast is more important to human perception than absolute intensities. Land and McCann's Retinex theory [Land 71] and the Cornsweet illusion [Corn 70] in Figure 1.3 both support the importance of local contrast on human perception over the underlying intensity values. This is why we model the local contrast in an image and disregard more spatially distant relationships. Additionally, operating globally between all parts of the image introduces a heavily constrained system that can become very difficult to satisfy. On the other hand, a purely local method only accounts for the relationships between neighboring pixels. Such methods do not consider the relationships between neighboring regions which can represent image features. These purely local methods would destroy relationships between similar, non-neighboring colors in a local region, for example the relationships between multiple patches of sky shining through the branches of the tree. With our method, the multiple patches of sky would be grouped into the same cluster and modified as a whole. The issues encountered by purely global or purely local solutions are avoided with our semi-local clustering. However, we note that the tradeoff between local and global operation can be controlled through the relative weighting in the spatial and chromatic dimensions in the clustering step.

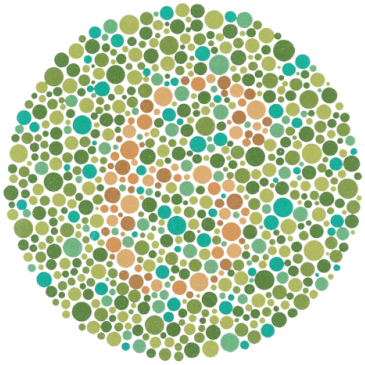
Our cluster-based method applied to the problems in the framework accomplishes our two goals. Our results show that our method is able to preserve local contrasts that were lost during the standard projection to the target space while remaining close to the natural colors of that projection, thereby maintaining naturalness. A benefit of our semi-local, cluster-based method is that our method avoids halo artifacts that arise in methods with local filtering operations. Also, by working with clusters, we are less susceptible to color ramping artifacts that could occur in gradient-based methods. Moreover, our main contribution is a single, coherent method that applies to all the applications in the framework: color to gray conversion, color gamut mapping, image optimization for color deficient viewers, and multispectral image fusion.

Our method has the potential for future applications in color image processing. As future work, our cluster-based method could be applied to other problems that fit within the framework such as tone mapping. Alternatively, our cluster-based mapping approach could be extended to apply to other related problems such as colorization or gamut expansion. We hope that the unified manner of our framework will inspire the development of general methods for all problems in the framework or the adaptation of existing single-problem methods to solve other problems in the framework.

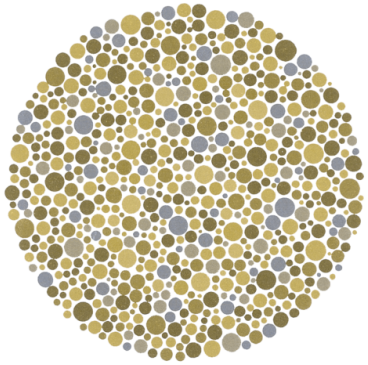
(a) color image (b) standard grayscale image (CIE L^*)

(c) original image (d) standard mapping to smaller gamut

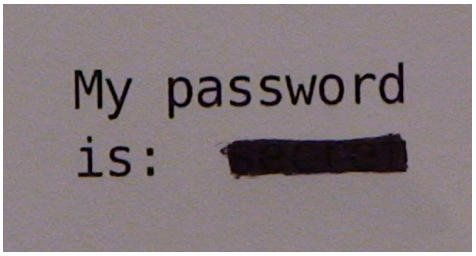
Figure 1.1: **Contrast lost during standard mapping.** During transformations from a larger source space to a smaller target space, information loss is inevitable. The strong color contrast between the red and blue elements in (a) is lost during a standard conversion to grayscale (b). The wrinkles in the hat in (c) are lost during the standard gamut clipping to a reduced space (d). *Original images courtesy of (a) Gooch et al. [Gooc 05] and (b) Kodak [Koda].*



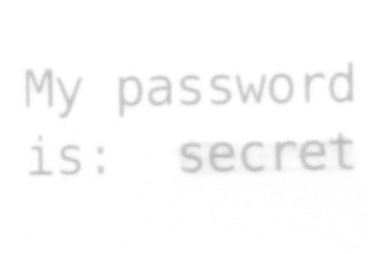
(a) color image



(b) simulated color deficient viewer

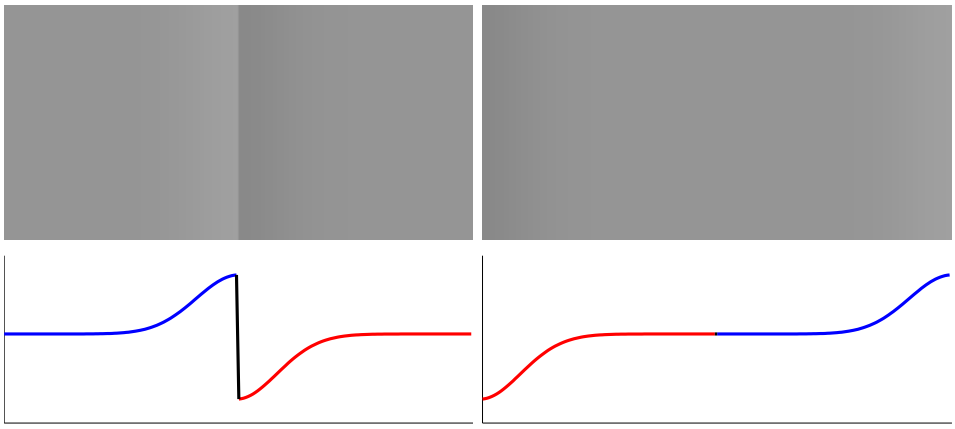


(c) visible image



(d) near-infrared image

Figure 1.2: **Contrast lost during standard mapping (more examples).** During transformations from a larger source space to a smaller target space, information loss is inevitable. When simulating in (b) how a color deficient viewer sees the color image (a), the chromatic contrast that makes the number '6' visible in (a) is lost. The secret password, captured with near-infrared light in (d), is hidden when only visible light is considered in the natural representation of a scene to a human viewer in (c). *Original image (a) courtesy of Wikipedia [Ishi].*



(a) Cornsweet illusion (b) signals switched

Figure 1.3: **Cornsweet Illusion.** The Cornsweet Illusion [Corn 70] (a) demonstrates the importance of local contrast in human perception. The illusion creates an apparent step edge due to the local contrast despite the fact that the intensity signals in blue and red indicate that the extreme ends of the signals are in fact the same intensity. This is evident in (b) when we put the ends of the signals next to each other by switching the order of the signals, placing the red signal on the right and the blue signal on the left.

Chapter 2

Background

This chapter contains some background information on topics related to this thesis. Most of this chapter is based on Reinhard et al.'s book *Color Imaging: Fundamentals and Applications* [Rein 08] to which the reader is referred for more detailed discussions on these topics.

2.1 Color Spaces

Color spaces have been developed to specify colors. They provide a mechanism with which we can quantify colors and decide when colors match. From the Wright and Guild color matching experiments, the color matching functions were established, leading to the CIE RGB and XYZ color spaces and the specification of colors using tristimulus values. Other color spaces have been developed for a variety of reasons including physical realizability of primaries, more efficient encoding, and perceptual uniformity. The XYZ primaries are imaginary, meaning light sources for these primaries do not exist that can be used to create a display device. Different RGB spaces have been created to represent data for different RGB devices. Colors are more efficiently encoded in sRGB space, which involves a nonlinear gamma encoding. CIE LAB and CIE LUV are perceptually uniform color spaces in which perceptual differences can be measured with Euclidean distances. The LMS color space approximates the cone responses of the human visual system. Depending on the task, one color space might be better to use over another.

2.1.1 Color Matching Functions

In the 1920's, Wright and Guild set up color matching experiments to determine the relative amounts of red, green, and blue lights needed to represent each monochromatic wavelength. In the experimental setup, lights are projected onto a wall divided into two halves representing a test field and a reference field. A monochromatic light of a particular wavelength is projected onto the reference field while red, green, and blue lights are projected

onto the test field. Human observers look at both fields and try to make the test field match the reference field by adjusting knobs for the red, green, and blue lights to control their contributions. Another set of red, green, and blue lights are also adjustable on the reference side; these lights correspond to negative contributions to the test field. These experiments resulted in the CIE 1931 color matching functions $\overline{r}(\lambda), \overline{g}(\lambda), \overline{b}(\lambda)$ shown in Figure 2.1. They describe how much of each red, green, and blue primary is needed to match a given wavelength.

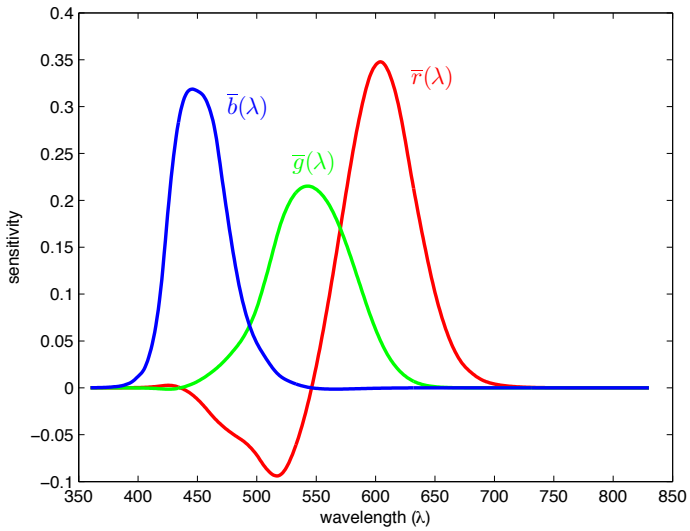


Figure 2.1: **RGB color matching functions.** These are the CIE 1931 2° color matching function $\overline{r}(\lambda), \overline{g}(\lambda), \overline{b}(\lambda)$ based on the Wright and Guild experiments.

Since the $\overline{r}(\lambda), \overline{g}(\lambda), \overline{b}(\lambda)$ color matching functions contain negative values, the CIE created another set of color matching functions which does not have negative values. These are the $\overline{x}(\lambda), \overline{y}(\lambda), \overline{z}(\lambda)$ color matching functions shown in Figure 2.2, and they are related to the $\overline{r}(\lambda), \overline{g}(\lambda), \overline{b}(\lambda)$ color matching functions by a linear transformation. These experiments were performed for a visual field of 2°, so the color matching functions are known as the CIE 1931 2° color matching functions.

At each wavelength λ, the color matching functions define the spectral tristimulus values that are the relative amounts of each primary needed

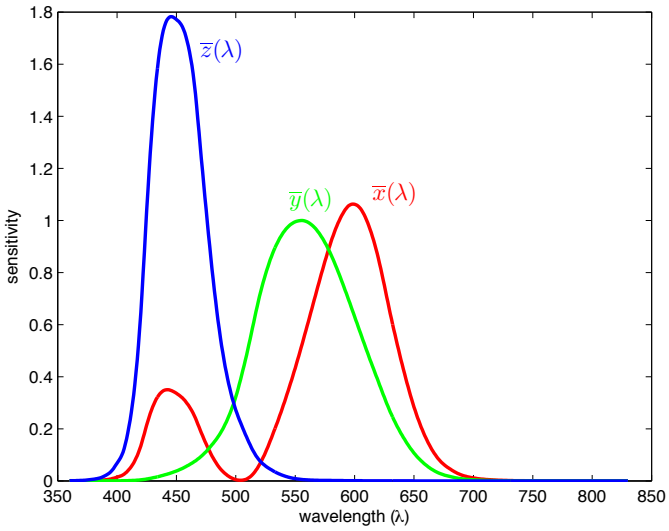


Figure 2.2: **XYZ color matching functions.** These are the CIE 1931 2° color matching functions $\overline{x}(\lambda), \overline{y}(\lambda), \overline{z}(\lambda)$ created from a linear transform of the $\overline{r}(\lambda), \overline{g}(\lambda), \overline{b}(\lambda)$ color matching functions.

to represent the monochromatic wavelength. Tristimulus values, RGB or XYZ, for a spectral response are calculated from their respective set of color matching functions by multiplying the spectral response by each color matching function and integrating. For a spectral response $I(\lambda)$, its RGB tristimulus values are calculated as

$$\begin{aligned} R &= \int_{\lambda} I(\lambda)\overline{r}(\lambda)d\lambda, \\ G &= \int_{\lambda} I(\lambda)\overline{g}(\lambda)d\lambda, \\ B &= \int_{\lambda} I(\lambda)\overline{b}(\lambda)d\lambda. \end{aligned} \tag{2.1}$$

Similarly, the XYZ tristimulus values for spectral response $I(\lambda)$ are calcu-

lated as

$$X = \int_{\lambda} I(\lambda)\overline{x}(\lambda)d\lambda, \tag{2.2}$$
$$Y = \int_{\lambda} I(\lambda)\overline{y}(\lambda)d\lambda,$$
$$Z = \int_{\lambda} I(\lambda)\overline{z}(\lambda)d\lambda.$$

2.1.2 XYZ

CIE XYZ is a device-independent color space. It is often used as an intermediate color space when transforming between other color spaces, especially device-dependent color spaces.

The XYZ color space arises from the $\overline{x}(\lambda), \overline{y}(\lambda), \overline{z}(\lambda)$ color matching functions, which are based on the Wright and Guild experiments. The calculation of XYZ tristimulus values from spectral responses is described in Section 2.1.1 and Equation 2.2. By design, the $\overline{y}(\lambda)$ color matching function is almost identical to the CIE 1924 2° photopic luminous efficiency curve $V(\lambda)$, which models the human eye's sensitivity to daylight. This means that CIE Y values are measures of luminance, or light weighted by the sensitivity of the human eye.

xy Chromaticity Coordinates

XYZ tristimulus values can be projected from three dimensional space to a two dimensional space representing just the chromatic information. This projection normalizes the XYZ values, yielding chromaticity coordinates x, y, and z.

$$x = \frac{X}{X+Y+Z}, \tag{2.3}$$
$$y = \frac{Y}{X+Y+Z},$$
$$z = \frac{Z}{X+Y+Z}.$$

The chromaticity coordinates sum to one, i.e. $x + y + z = 1$, so the z coordinate is redundant when the other two are known. The xy chromaticity diagram consists of the x and y coordinates and is shown in Figure 2.3. The horseshoe-shaped boundary is called the spectral locus, and it corresponds to the monochromatic wavelengths in the visible range. The boundary at

the bottom of the horseshoe is called the purple line and does not correspond to spectral colors. Since the xy coordinates alone do not contain luminance information, specifying the Y value along with x and y is necessary to convert chromaticities back to XYZ tristimulus values.

$$\begin{aligned} X &= \frac{x}{y}Y, \\ Y &= Y, \\ Z &= \frac{z}{y}Y. \end{aligned} \tag{2.4}$$

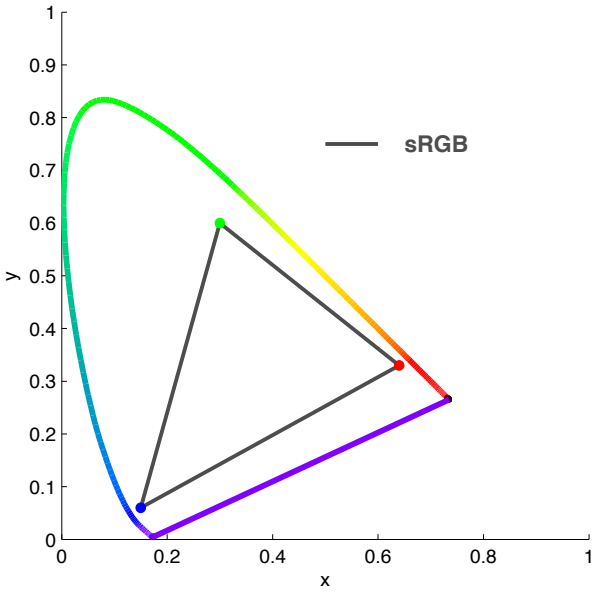


Figure 2.3: xy **chromaticity diagram with sRGB gamut.**

Often, chromaticities are described by their xy chromaticity coordinates. For example, illuminants or primaries of a device are frequently specified in xy coordinates. By plotting the primaries of a device in the chromaticity diagram and drawing lines between the primaries, the resulting shape encloses all the chromaticities that are reproducible by that device. A gamut is a set of colors that typically describes what colors a device can reproduce. The sRGB color space, defined in Section 2.1.3, is used to describe the gamut of many of today's standard display devices. The triangle for the sRGB gamut

is drawn on the chromaticity diagram in Figure 2.3. This is one way to convey the gamut of a device. However, the gamut is not fully described by this 2D diagram as a gamut is a 3D volume which needs to be specified in the luminance dimension as well.

2.1.3 RGB

Conventional display and capture devices have red, green, and blue primaries. RGB color spaces are common for representing image data for these devices. Since devices have different primaries and white points, different RGB color spaces have been developed. Each RGB color space is defined by its primaries, white point, and a transfer function defining the gamma encoding curve.

sRGB

The sRGB color space is a common color space used for conventional devices and the Internet. Its primaries and white point are specified by the ITU-R BT.709 standard [Recob] (abbreviated as Rec. 709) and are listed in Table 2.1. These primaries are also plotted in the chromaticity diagram in Figure 2.3.

Table 2.1: **sRGB primaries and white point.** The sRGB primaries and white point (W) are specified by ITU-R BT.709 [Recob].

	R	G	B	W
x	0.6400	0.3000	0.1500	0.3127
y	0.3300	0.6000	0.0600	0.3290

The transformation from XYZ to sRGB first involves a matrix multiplication to transform into linear RGB values which is then followed by a nonlinear gamma encoding function. The linear RGB values are calculated from XYZ values in $[0, 1]$ by a matrix multiplication:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.2405 & -1.5371 & -0.4985 \\ -0.9693 & 1.8706 & 0.0416 \\ 0.0556 & -0.2040 & 1.0572 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.5}$$

The matrix multiplication can result in negative numbers. Therefore, the linear RGB values are clipped to the range $[0, 1]$. Then, a nonlinear transfer

function is applied to the linear RGB values, resulting in the nonlinear $R'G'B'$ values of sRGB space. Equation 2.6 shows this calculation for R', but G' and B' are defined similarly.

$$R' = \begin{cases} 12.92R, & R \leq 0.0031308, \\ (1+a)R^{\frac{1}{2.4}} - a, & R > 0.0031308, \end{cases} \tag{2.6}$$

where $a = 0.055$.

The transformation from sRGB values to XYZ values first linearizes the $R'G'B'$ values and then converts the linear RGB values to XYZ. Again, Equation 2.7 applies the inverse transfer function to R', but similar calculations are performed for G' and B'.

$$R = \begin{cases} \frac{R'}{12.92}, & R' \leq 0.04045, \\ (\frac{R'+a}{1+a})^{2.4}, & R' > 0.04045, \end{cases} \tag{2.7}$$

where $a = 0.055$. The linear RGB values are then converted to XYZ.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4124 & 0.3576 & 0.1805 \\ 0.2126 & .07152 & 0.0722 \\ 0.0193 & .01192 & 0.9505 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{2.8}$$

Luma

Luma is a weighted sum of nonlinear $R'G'B'$ components. The weights can be determined by various specifications. For example, the Rec. 709 [Recob] coefficients define luma Y' as

$$Y' = 0.2126R' + 0.7152G' + 0.0722B'. \tag{2.9}$$

The Rec. 601 [Recoa] coefficients define luma as

$$Y' = 0.299R' + 0.587G' + 0.114B'. \tag{2.10}$$

2.1.4 $L^*a^*b^*$

CIE LAB is a perceptually uniform color opponent space. Being a color opponent space means that it has two chromatic axes, a^* and b^*, roughly corresponding to red-green opponency and blue-yellow opponency, respectively. It also has an achromatic dimension, CIE L^*, which is the lightness axis. It is calculated from a cube root function of luminance Y, so it better approximates human perception than the linear luminance encoding of

Y. In perceptually uniform spaces, it is possible to use Euclidean distance between colors as a measure of perceptual difference. Comparing Euclidean distances roughly corresponds to comparing perceptual differences between colors. One unit in $L^*a^*b^*$ space roughly corresponds to one JND, or just-noticeable-difference.

To convert XYZ values to $L^*a^*b^*$ space, the white point also needs to be specified. L^* values range from $[0, 100]$ with a value of 100 corresponding to the white point. The a^* and b^* axes are not constrained. Given XYZ values and the white point (X_n, Y_n, Z_n) , the $L^*a^*b^*$ values are defined as

$$\begin{bmatrix} L^* \\ a^* \\ b^* \end{bmatrix} = \begin{bmatrix} 0 & 116 & 0 & -16 \\ 500 & -500 & 0 & 0 \\ 0 & 200 & -200 & 0 \end{bmatrix} \begin{bmatrix} f(X/X_n) \\ f(Y/Y_n) \\ f(Z/Z_n) \\ 1 \end{bmatrix} \tag{2.11}$$

where the function $f()$ is

$$f(r) = \begin{cases} \sqrt[3]{r}, & r > 0.008856, \\ 7.787r + \frac{16}{116}, & r \leq 0.008856. \end{cases} \tag{2.12}$$

Given the white point (X_n, Y_n, Z_n), $L^*a^*b^*$ values are transformed back into XYZ values by

$$\begin{aligned} X &= X_n \begin{cases} (\frac{L^*}{116} + \frac{a^*}{500} + \frac{16}{116})^3 & L^* > 7.9996, \\ \frac{1}{7.787}(\frac{L^*}{116} + \frac{a^*}{500}) & L^* \leq 7.9996, \end{cases} \\ Y &= Y_n \begin{cases} (\frac{L^*}{116} + \frac{16}{116})^3 & L^* > 7.9996, \\ \frac{1}{7.787}(\frac{L^*}{116}) & L^* \leq 7.9996, \end{cases} \\ Z &= Z_n \begin{cases} (\frac{L^*}{116} - \frac{b^*}{200} + \frac{16}{116})^3 & L^* > 7.9996, \\ \frac{1}{7.787}(\frac{L^*}{116} - \frac{b^*}{200}) & L^* \leq 7.9996. \end{cases} \end{aligned} \tag{2.13}$$

The color difference metric ΔE^*_{ab} is the Euclidean distance between two points in $L^*a^*b^*$ space.

$$\Delta E^*_{ab} = \sqrt{(\Delta L^*)^2 + (\Delta a^*)^2 + (\Delta b^*)^2} \tag{2.14}$$

Even though $L^*a^*b^*$ space is classified as a perceptually uniform space, it is not perfectly linear with respect to human perception. It is only approximately linear. For large color differences, the color difference metric breaks down.

$L^*C^*h_{ab}$

The polar representation of $L^*a^*b^*$ space is $L^*C^*h_{ab}$ space. The L^* axis remains the same, but the a^*b^* plane is represented in polar coordinates: chroma C^* and hue angle h_{ab}. In polar space, planes of constant hue can be selected for a particular hue angle. The two dimensions of a constant hue plane are lightness L^* and chroma C^*. The conversion from $L^*a^*b^*$ is

$$\begin{aligned} L^*_{ab} &= L^*, \\ C^*_{ab} &= \sqrt{a^{*2} + b^{*2}}, \\ h_{ab} &= \tan^{-1}(\frac{b^*}{a^*}). \end{aligned} \tag{2.15}$$

and the conversion back to $L^*a^*b^*$ is

$$\begin{aligned} L^* &= L^*, \\ a^* &= C^* \cos(h_{ab}), \\ b^* &= C^* \sin(h_{ab}). \end{aligned} \tag{2.16}$$

2.1.5 $L^*u^*v^*$

CIE LUV space is another perceptually uniform color opponent space. It was developed around the same time as $L^*a^*b^*$, and both spaces were standardized by the CIE. Typically, $L^*a^*b^*$ space is used more often in the printing and materials industry while $L^*u^*v^*$ space is used more often in the display industry. In $L^*u^*v^*$ space, the u^* and v^* axes represent red-green opponency and blue-yellow opponency, respectively. The lightness axis L^* is the same as in $L^*a^*b^*$ space. Given the white point (X_n, Y_n, Z_n), the transformation from XYZ to $L^*u^*v^*$ is

$$\begin{aligned} L^* &= \begin{cases} (\frac{29}{3})^3(\frac{Y}{Y_n}), & \frac{Y}{Y_n} \leq (\frac{6}{29})^3, \\ 116(\frac{Y}{Y_n})^{\frac{1}{3}} - 16, & \frac{Y}{Y_n} > (\frac{6}{29})^3, \end{cases} \\ u^* &= 13L^*(u' - u'_n), \\ v^* &= 13L^*(v' - v'_n), \end{aligned} \tag{2.17}$$

where u' and v' are

$$\begin{aligned} u' &= \frac{4X}{X + 15Y + 3Z}, \\ v' &= \frac{9Y}{X + 15Y + 3Z}, \\ u'_n &= \frac{4X_n}{X_n + 15Y_n + 3Z_n}, \\ v'_n &= \frac{9Y_n}{X_n + 15Y_n + 3Z_n}. \end{aligned} \tag{2.18}$$

The reverse transformation from $L^*u^*v^*$ to XYZ starts with the calculation of u'_n and v'_n from the white point (X_n, Y_n, Z_n) according to Equation 2.18. XYZ values are then given by

$$\begin{aligned} u' &= \frac{u^*}{13L^*} + u'_n, \\ v' &= \frac{v^*}{13L^*} + v'_n, \\ X &= Y(\frac{9u'}{4v'}), \\ Y &= \begin{cases} Y_n \cdot L^*(\frac{3}{29})^3, & L^* \leq 8, \\ Y_n(\frac{L^*+16}{116})^3, & L^* > 8, \end{cases} \\ Z &= Y(\frac{12 - 3u' - 20v'}{4v'}). \end{aligned} \tag{2.19}$$

The color difference metric ΔE^*_{uv} is Euclidean distance in $L^*u^*v^*$ space. Since $L^*u^*v^*$ is perceptually uniform, Euclidean distance can be used as a measure of perceptual difference.

$$\Delta E^*_{uv} = \sqrt{(\Delta L^*)^2 + (\Delta u^*)^2 + (\Delta v^*)^2} \tag{2.20}$$

The polar representation of $L^*u^*v^*$ space is $L^*C^*h_{uv}$ space. It is calculated exactly as $L^*C^*h_{ab}$ space is calculated for $L^*a^*b^*$ space with u^* and v^* replacing a^* and b^*. L^* stays the same, C^* is chroma, and h_{uv} is the hue angle.

2.1.6 LMS

The LMS color space is based on the three types of cones in the human visual system. The cones are distinguished by their response functions, which correspond to long (L), medium (M), and short (S) wavelengths. The

LMS color space is defined by a linear transformation from CIE XYZ space where the resulting LMS response functions approximate the cone response functions of the human visual system. A common transformation uses the Hunt-Pointer-Estevez primaries to approximate the cone responses. The conversion from XYZ to LMS using the Hunt-Pointer-Estevez primaries and assuming an equal energy illuminant is given by

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.38971 & 0.68898 & -0.07868 \\ -0.22981 & 1.18340 & 0.04641 \\ 0.00000 & 0.00000 & 1.00000 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.21}$$

The transformation from LMS to XYZ is given by

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 1.91020 & -1.11212 & 0.20191 \\ 0.37095 & 0.62905 & -0.00001 \\ 0.00000 & 0.00000 & 1.00000 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix}. \tag{2.22}$$

This transformation results in the relative response functions for the LMS cone space, which are plotted in Figure 2.4.

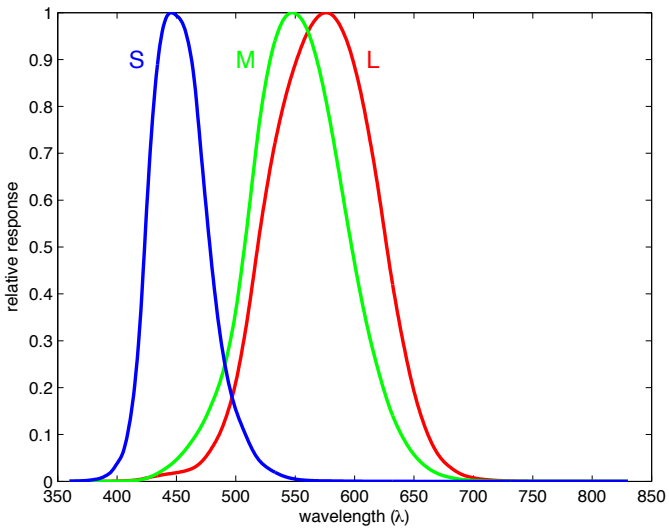


Figure 2.4: **Relative responses for LMS cone space.** These are the relative response functions for the LMS cone space defined by the Hunt-Pointer-Estevez cone primaries.

2.2 Human Vision

The human visual system has different components that affect how we perceive light, color, and contrast. The rods and cones are the low-level receptors in the human eye that sense light. Further processing is done by the visual system to convert the sensed signal into the visual sensation of light and color. Local contrast plays an important role in our perception of stimuli. This is evident from effects such as simultaneous contrast. To measure when the contrast between two stimuli is detected, the concept of the just noticeable difference (JND) was developed.

2.2.1 Rods and Cones

When light enters the human eye, it is projected onto the retina. The retina contains two types of photoreceptors called rods and cones. Rods are active during low light conditions, or scotopic conditions, while cones are active during bright light conditions, or photopic conditions. Both rods and cones are active in the mesopic range which occurs between scotopic and photopic conditions. There are three types of cones which collectively are responsible for color vision.

Rods and cones contain different light-sensitive pigments. Rods contain rhodopsin, which is sensitive over a wide range of wavelengths with a peak spectral sensitivity at 496 nm. The three cone types have different pigments, or opsins, with different spectral sensitivity functions. Their peak spectral sensitivities are around 440 nm, 545 nm, and 565 nm. Respectively, these cone types are referred to as the long wavelength (L), medium wavelength (M), and short wavelength (S) cones. The signals from these three types of cones form the base for how the human visual system perceives color.

2.2.2 Dual-Process Theory

The Young-Helmholtz trichromatic theory predicts that colors can be formed from a mixture of three primary colors. This is supported by the fact that there are three types of cones that have three different peak sensitivities.

Opponent-process theory predicts that there is a red-green channel, a blue-yellow channel, and an achromatic channel. The colors red and green are paired together in a single dimension. Since red is on the positive side and green is on the negative side with gray in the middle, a mixture of red and green does not exist. Similarly, mixtures of blue and yellow do not exist. These are opponent colors, and they explain visual phenomenon such

as color blindness. It explains how color blindness due to the disability of one cone affects pairs of colors in either the red-green channel or the yellow-blue channel.

The dual-process theory combines the trichromatic theory with the opponent-process theory in a unified, multi-stage process. First, trichromatic theory holds when light is detected by the three cone types. Then, opponent-process theory applies to the next level of processing where the signals from the cones are compared and color opponent signals are formed.

2.2.3 Simultaneous Contrast

Simultaneous contrast is a perceptual effect in which the perceived color of a stimulus changes depending on its background. In Figure 2.5, the gray patches in the center have the same intensity, but the white and black backgrounds make the gray patch on the white background look darker than the gray patch on the black background. This phenomenon also applies to color and is known as simultaneous color contrast. The perceived color of a gray patch is affected by the color of its background. A gray patch would appear tinted with the background color's complementary hue. With simultaneous contrast and simultaneous color contrast, the perceived color of a stimulus is altered by what is next to it.

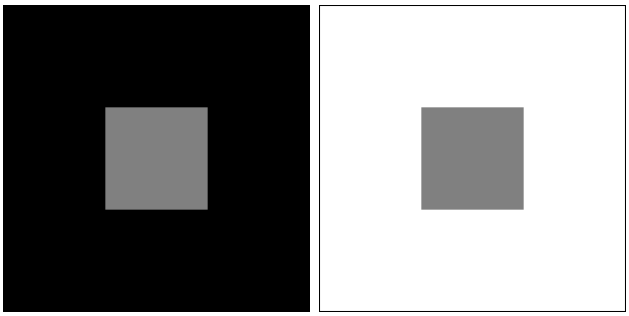


Figure 2.5: **Simultaneous contrast.** Although the gray patches are the same intensity, the gray patch on the white background looks darker. The background changes the perceived intensity of the patches.

2.2.4 Just Noticeable Difference

A just noticeable difference, or JND, is the smallest intensity difference between two stimuli at which the stimuli are detected to be different. This

intensity difference ΔI varies with the background intensity I to which the observer is adapted. Weber's Law states that ΔI is proportional to the background intensity [Blac 72, Pire 62]. This relationship is

$$\frac{\Delta I}{I} = k \tag{2.23}$$

where k is the Weber constant. Weber's Law holds for a wide range of background intensities but not for extreme dark or bright conditions. The just noticeable difference is often used as a threshold to tell whether or not an observer will be able to distinguish between two stimuli.

2.3 Gamma Encoding and Gamma Correction

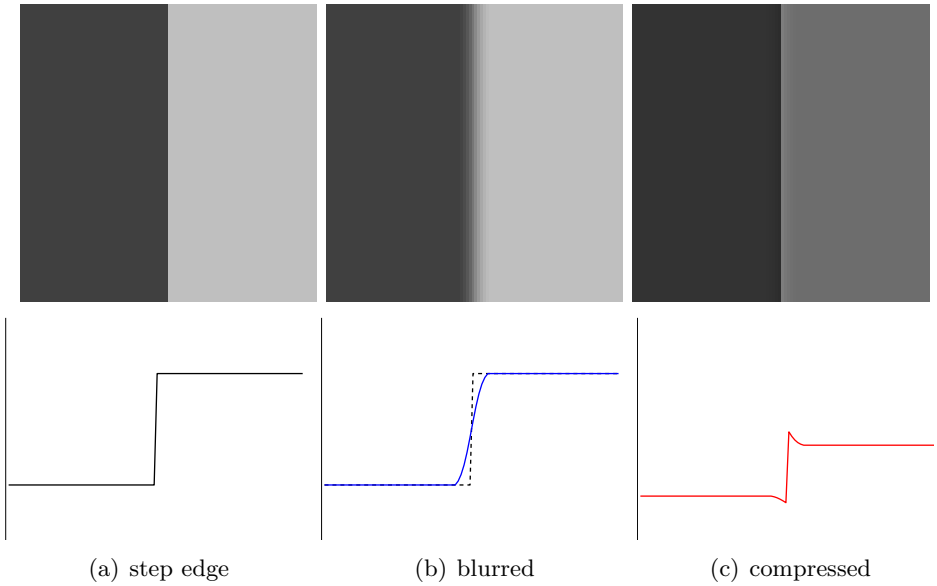
Typically, 8-bit pixel values in $[0, 255]$ are sent to the display. These RGB pixel values are usually nonlinearly encoded, such as in sRGB. The sRGB specification applies a nonlinear transfer function to linear RGB values as discussed in Section 2.1.3. Essentially, this transfer function encodes the linear values with a gamma of approximately $\frac{1}{2.4}$. Gamma encoding allows for more efficient allocation of the available bit depth according to human perception. It gives more bit depth to the darker luminances and less to the lighter luminances. The human visual system is more sensitive to changes in the darker regions than changes in the lighter regions. With a linear encoding, humans would be able to see quantization errors in the dark regions. The gamma encoding aims to keep quantization errors below visible thresholds. The nonlinear transfer function that performs gamma encoding also works as gamma correction.

Gamma correction is needed when sending input to displays that have a nonlinear output response, such as CRTs. For CRTs, the output luminance is related to the input voltage by a power law with exponent γ. A typical display gamma is around $\gamma = 2.2$. Gamma correction applies the inverse of the display curve to the input data before sending the signal to the display. This results in linear output luminances from the display. Since sRGB values are already gamma encoded, they can be sent directly to the display. To display linear luminance values as linear output luminances, the input should be gamma corrected. For a display with $\gamma = 2.2$, this means raising the data to the power $\frac{1}{2.2}$.

2.4 Halo Artifacts

Halo artifacts can occur when local filtering techniques are used. Halos are bands of brightness or darkness on either side of a step edge such that the intensity immediately next to the edge overshoots or undershoots the higher and lower intensities of the step edge, respectively. Halos arise because the local average computed by filtering can be very different from the pixel being filtered when the filter window lies across a strong edge. For example, when blurring an image, such as the step edge in Figure 2.6(a), with a Gaussian filter, the blurred values near the step edge are different from the underlying pixel values. When using the filtered values as local average estimates in a compression function, halo artifacts occur where the filtered values are very different from the pixels being filtered, as shown in Figure 2.6(b). In Figure 2.6(c), the edge is compressed using $L_d = \frac{L}{1+L_b}$ where L is the original image intensity, L_b is the Gaussian filtered value, and L_d is the output intensity. This is similar to the compression performed in Reinhard et al.'s photographic tone mapping operator [Rein 02]. To reduce the problems with filtering over strong contrasts, edge-aware filtering is often performed to minimize halo artifacts.

The edge profile in Figure 2.6(c) is similar in shape to the Cornsweet Illusion shown in Chapter 1, which created the illusion of a step edge. This effect can be successfully used in tone mapping applications. Smith et al. [Smit 06] add the Cornsweet profile to edges in order to increase their apparent contrast without requiring the full dynamic range of the edge. However, if over-exaggerated, the profiles look like halo artifacts. Trentacoste et al. [Tren 12] investigate the conditions under which adding Cornsweet profiles goes from enhancement to artifact, depending on the scale and magnitude of the Cornsweet profile.



(a) step edge (b) blurred (c) compressed

Figure 2.6: **Cause of halo artifacts.** Halo artifacts are common in local filtering techniques, and they arise when filtered values differ from their underlying pixel values due to filtering over strong edges. Here, images of an edge are shown with corresponding intensity profiles below. When the step edge in (a) is blurred by a Gaussian filter, the blurred values (blue) near the edge differ from the original edge intensities (black) shown in (b). Using the blurred values as local average estimates in a compression function causes halos in (c).

Chapter 3

Related Work

Each of the specific applications to which we apply our n to m dimensional mapping method has its own body of related work. Work on color to gray conversion focuses on converting 3D color images to a single grayscale dimension. Gamut mapping transforms an image from a source gamut in $\mathbb{R}^n$ to target gamut in $\mathbb{R}^m$, where n and m can be equal. Image optimization for color deficient viewers recolors images only using colors on the 2D surface that defines the scope of colors distinguishable by the color deficient viewer. Multispectral or multiprimary image fusion focuses on compressing n spectral bands, where $n > 3$, into a tristimulus image. Other related work includes more general problems of dimensionality reduction ($n > m$) and dimensionality expansion ($m > n$).

Most methods aim to preserve contrast lost during the transformation from one space in $\mathbb{R}^n$ to another space in $\mathbb{R}^m$. Furthermore, many methods can be classified as local or global. Our method focuses on enhancing local contrast and relinquishes the need for a global mapping, thereby giving us more flexibility to produce a natural looking result. Some methods use local filtering techniques or work with pixel level gradients which could lead to halo artifacts or unwanted color ramps in solid color regions. Our semi-local method avoids these artifacts by modifying clusters of pixels, or whole regions, at a time.

3.1 Color To Gray

Contrast preserving mappings from color images to grayscale have been the subject of a lot of research in recent years. The proposals include pixel level optimizations, global methods, local filtering methods, and gradient-based methods.

3.1.1 Pixel Level Optimization

Optimization methods for the color to gray problem try to solve for gray values that best portray color differences in the image. Gooch et al.'s [Gooc 05]

objective function tries to match grayscale differences between pixel pairs to target differences. Since their local method solves for the grayscale value of every pixel and considers all pixel pairs in the image, it is computationally expensive. Our work is most similar to Gooch et al. in that we also match grayscale differences to target differences. However, instead of operating at the pixel level, our semi-local method solves for the graylevels of clusters and considers only spatially close cluster pairs, making our optimization faster and less constrained.

3.1.2 Global Methods

A variety of methods solve for a global mapping of colors to grayscale values. Rasche et al.'s [Rasc 05b] and Cui et al.'s [Cui 10] approaches are based on multidimensional scaling (MDS). The optimization method of Rasche et al. [Rasc 05b] seeks a global mapping that preserves contrast by keeping grayscale differences proportional to color differences. Cui et al.'s [Cui 10] method is based on ISOMAP, a special case of MDS. Kuhn et al. [Kuhn 08b] solve for a global mapping by setting up a mass spring system based on perceptual color differences. The system reaches an optimal set of gray values at equilibrium. The Decolorize algorithm by Grundland and Dodgson [Grun 07] computes chrominance information by projecting data onto an image dependent axis that contains the highest loss in chrominance contrast. The methods of Kim et al. [Kim 09] and Zhao and Tamimi [Zhao 10] add chrominance information to the luminance channel. Kim et al. [Kim 09] add a function of the C and H channels to the L channel in LCH space. Zhao and Tamimi [Zhao 10] add the a^* and b^* chromatic channels to L^* in the Fourier domain. While global mappings provide a consistent solution across the image, a color could be mapped to a gray value that is unrealistically far from its standard gray value. We adopt a local method that allows the optimization more freedom to remain close to natural gray values while exhibiting local contrast. We uphold that modifying local contrast is more important than fulfilling a consistent mapping since maintaining local contrast is more important to human perception than maintaining a strict mapping correlated with the underlying absolute intensities.

3.1.3 Local Filtering Methods

Some methods enhance local contrast by employing local filtering techniques. However, this could lead to halo artifacts. Bala and Eschbach [Bala 04] apply a high pass filter to their chrominance channels and add the high fre-

quencies back into the luminance channel. The high pass filtering causes the output to be enhanced only at the edges whereas our method modifies whole regions. Smith et al. [Smit 08] aim to convert a color image into a perceptually accurate grayscale image with a two step process. The first step assigns a perceptually accurate global mapping by computing each color's apparent lightness value based on the Helmholtz-Kohlrausch (H-K) effect. The H-K effect is a phenomenon in which a more saturated color appears lighter than a less saturated color with the same luminance. The second step enhances local contrast with a multiscale unsharp masking technique. Ranked as the top color to gray method on accuracy in a experimental study by Cadik [Cadi 08], they accomplish their goal of creating perceptually accurate grayscale images. Applying the H-K effect yields perceptual accuracy, but our goal is to enhance contrast between different colors even if the colors have the same apparent graylevel. Furthermore, their unsharp masking step and likewise Bala and Eschbach's [Bala 04] high pass filtering method introduce halos around large regions of homogenous color. Our method avoids filtering operations and instead modifies regions of pixels as groups.

3.1.4 Gradient-Based Methods

Gradient-based methods create a gradient field that best captures the contrast in the color image, and then, the representative grayscale image is obtained by either solving a Poisson equation or integrating the gradient field. Socolinsky and Wolff's method [Soco 02], though developed for creating grayscale images from multispectral images, is applied to converting color to gray. Gradients in three color channels are captured in a 2×2 structure tensor, and the contrast of the color image is represented by the largest eigenvector and eigenvalue. These eigenvectors and eigenvalues make up the gradient field. Drew et al. [Drew 09] build upon Socolinsky and Wolff's method [Soco 02], improving the gradient's sign determination by picking the sign that minimizes curl. Alsam and Drew's method [Alsa 08] is fast because it replaces the eigenvalue decomposition with a max gradient calculation. A different gradient-based method by Neumann et al. [Neum 07] calculates the gradient field based on the Coloroid system. Then, the gradient field is projected to the nearest consistent gradient field allowing them to integrate to obtain the grayscale image. While gradient methods provide an elegant solution to the color to gray problem, their focus is on the relationships between pixels since gradients are calculated from pixel neighborhoods. In contrast, our method uses differences between clusters and shifts the focus to gradients between neighboring areas as a whole.

3.1.5 Invertible Methods

Alternatively, DeQueiroz and Braun [DeQu 06] encode chrominance information as high frequency texture in a grayscale image. Their color to gray transformation is invertible, and they decode the chromatic channels using a wavelet transform.

3.2 Gamut Mapping

The goal of gamut mapping methods is to simultaneously preserve original contrasts and colors. To this end, many types of methods have been developed, and overviews and surveys can be found in work by Morovic and Luo [Moro 01] and Morovic [Moro 08], on which this review is partially based. Gamut mapping algorithms can be divided into global methods and local methods. Global methods, or pointwise methods, create a one-to-one mapping that maps each point in the source gamut to a point in the target gamut. Local, or spatial, methods take into account local neighborhoods around pixels and have the flexibility of a one-to-many mapping. Spatial gamut mapping methods can be further divided into methods that decompose images into bands and successively add detail layers, retinex based methods, and optimization methods based on image difference metrics.

3.2.1 Global Methods

Global gamut mapping algorithms can be categorized as clipping or compression methods. During gamut clipping, out-of-gamut points are projected to the gamut boundary while in-gamut points are unmodified. This leads to a loss of detail in the out-of-gamut regions. On the other hand, gamut compression methods remap all colors to maintain the relationships between colors rather than the colors themselves. For this reason, they are better at preserving detail but at the expense of inaccurate colors. The CIE Guidelines for evaluating gamut mapping algorithms [CIE 04] require testing with two standard global methods: Hue-angle Preserving Minimum ΔE_{ab} (HPMINDE) clipping and Sigmoidal Gaussian Cusp Knee (SGCK) compression. HPMINDE maps out-of-gamut points to their closest points on the target gamut boundary within constant hue angle planes. SGCK compresses lightness and chroma along constant hue lines pointed towards the lightness value corresponding to a cusp, or max chroma value, in the target gamut. Global methods must choose to tradeoff contrast preservation and color accuracy uniformly across the image, but spatial methods, such as ours, aim for a

better tradeoff by locally determining the tradeoff for each neighborhood.

3.2.2 Spatial Methods

With their one-to-many mappings, spatial methods have more flexibility than global methods. Clipping methods suffer from detail loss in out-of-gamut regions while compression methods can preserve those details at the cost of overall contrast loss [Moro 08]. Spatial methods compute mapped colors for each pixel based on each pixel's local neighborhood. By relaxing the constraint for a global mapping, several instances of the same color can map to different target colors depending on each instance's neighborhood; this gives a color in the source gamut more flexibility to move within the target gamut in order to preserve local detail while maintaining global contrast. Spatial methods try to achieve a better balance between color accuracy and detail preservation, the two competing goals in clipping and compression methods [Bonn 06]. Although spatial methods are more computationally complex than global methods, psychophysical experiments by Bonnier et al. [Bonn 06] show that results from spatial methods are preferred over results from global methods.

3.2.3 Methods That Add Detail Layers

One class of spatial gamut mapping algorithms involves decomposing an image into bands and adding detail layers. The first spatial method by Meyer and Barth [Meye 89] decomposes lightness into low and high pass bands using a Gaussian filter. Then, the low pass lightness image is compressed and the high pass details are added to it. The chroma channel is also compressed while hue is not modified. Finally, gamut clipping is applied to make sure the image is within the target gamut. Balasubramanian et al. [Bala 00] build on Meyer and Barth by computing their detail image based on the difference between original and mapped images. Morovic and Wang [Moro 03] decompose an image into multiple levels of frequency bands using mean filtering and successively add levels of details. Decomposing signals with Gaussian and mean filtering leads to halo artifacts, so subsequent work tries to preserve edges while reducing halos by using a bilateral filter [Zoll 07], computing weight maps [Faru 07], or applying an edge preserving filter to a compression map [Kola 07]. After a bilateral filter based decomposition, Bonnier et al. [Bonn 07, Bonn 08b] propose adaptive clipping and adaptive compression methods to merge the bands in ways that preserve local variations. These spatial methods are good at preserving details, but

the prevailing issue with this category of spatial methods is the tendency to produce halo artifacts. Although mitigated by the use of edge preserving filters, the result might not be entirely halo free. In our method, we avoid halos by working with clusters of data rather than spatially filtered pixels.

3.2.4 Retinex-Based Method

McCann [McCa 99] developed a spatial gamut mapping algorithm based on Retinex theory [Land 71]. Retinex theory proposes that local contrasts at multiple scales influence human visual perception more than the underlying luminance values themselves. Therefore, McCann's method creates multiresolution pyramids of both original and mapped images and optimizes to preserve ratios between spatial neighbors at all scales. This method preserves local contrast of the input image since ratios are reinforced at each level, but parallel processing of color channels leads to color shifts [Bonn 08a].

3.2.5 Optimization Methods

Optimization methods form another class of spatial gamut mapping algorithms. The first optimization approach by Nakauchi et al. [Naka 99] finds a mapped image within the target gamut that is perceptually closest to the original image as determined by a perceptual image difference metric. Kimmel et al. [Kimm 05] formulate spatial gamut mapping as a quadratic programming problem in which they minimize their perceptual difference metric. Giesen et al. [Gies 06] solve a constrained optimization using hard constraints to preserve detail, hue, achromatic colors, mapping continuity, and soft constraints to minimize distortion. Their method solves for a global mapping but might have potential as a spatial method [Moro 08] if spatial pairs were considered instead of pairs between all image gamut colors. Alsam and Farup [Alsa 09b] optimize for the gamut mapped image whose gradients best match the original image gradients under the constraint that colors do not change hue. Unfortunately, the method of Nakauchi et al. is highly dependent on the perceptual difference metric which in practice might not be the best choice [Moro 08]. The issue that persists throughout classes of spatial gamut mapping algorithms is the occurrence of halo artifacts. The methods of Kimmel et al. and Alsam and Farup are both susceptible to halos, though artifacts are reduced by blending between solutions in Kimmel et al.'s method or by limiting the number of iterations in Alsam and Farup's method.

3.3 Image Optimization for Color Deficient Viewers

When optimizing images for color deficient viewers, the goal is to enable the color deficient viewer to see the contrast that a normal viewer would see. In order to do that, we first need to be able to simulate what a color deficient viewer would see, and there are a few ways to do that. When creating new content, guidelines exist to help designers choose appropriate colors for different viewers. For existing content, daltonization methods recolor images for color deficient viewers. These methods solve for new colors using optimization techniques or other algorithms. Additionally, there are products designed to assist color deficient viewers such as glasses or iPhone apps.

3.3.1 Simulation

Given a color, simulation methods [Meye 88, Kond 90, Walr 97, Bret 97, Mach 09] emulate the human visual system of a color deficient viewer to predict what color the viewer most likely sees. In our method, we use Brettel et al. [Bret 97] to simulate the three types of dichromacy. First, colors are converted to LMS cone space. In LMS space, the surface of colors seen by a dichromat consists of two planes hinged at the neutral axis where the orientation of planes depends on the type of dichromacy. Then, the LMS values are projected onto this surface in the direction of the missing cone's axis, simulating the viewer's lack of sensitivity for that cone. Finally, these projected colors are converted back to RGB as the simulated output.

3.3.2 Guidelines

Some resources offer guidelines or tools to assist designers with choosing appropriate colors for different viewers. The World Wide Web Consortium's Web Content Accessibility Guidelines contain color visibility criteria for making websites accessible to different users [Chis 99]. Walraven and Alferdinck [Walr 97] create an editor that detects colors that look the same for color deficient viewers and allows users to pick better colors. The editor can also provide a default palette of distinguishable colors. Rigden [Rigd 99] and Viénot et al. [Vien 99] both provide reduced color palettes that show how a dichromat would see a normal color palette. The normal color palette could be replaced by the reduced palette to approximate dichromatic vision so that designers can check the readability of their content. These guidelines

and tools are useful when creating new content but they take a lot of effort on the designer's part.

3.3.3 Recoloring Methods

Recoloring methods are often referred to as daltonization. This term is named after John Dalton, a color deficient scientist who published the first paper on color blindness in which he cited his own condition. Optimization methods solve for new colors that preserve the distance relationships between the original colors. Ichikawa et al. assign fitness values to colors based on how well distances between colors are preserved. They use a genetic algorithm to recolor webpages with the best fit colors [Ichi 03] and later extend their work to images [Ichi 04]. In Rasche et al.'s earlier work [Rasc 05a], they solve for the linear transformation that best maintains color difference ratios. However, the transformation could produce colors outside the viewer's gamut. Later, they abandon the limiting notion of a linear transform in favor of a multidimensional scaling formulation yielding a nonlinear global mapping to colors within a viewer's gamut [Rasc 05b]. Wakita and Shimamura's [Waki 05] SmartColor method recolors documents based on the author's constraints dictating contrast preservation and naturalness. Jefferson and Harvey [Jeff 06] optimize for new colors that preserve target distances based on the recommended W3C criteria [Chis 99]. Similar to Rasche et al., whose methods [Rasc 05a, Rasc 05b] are extensions of their color to gray work, Kuhn et al. also apply their mass spring optimization for color to gray [Kuhn 08b] to recoloring images for color deficient viewers [Kuhn 08a]. Tanaka et al. [Tana 10] solve for new lightness values that preserve original contrasts for dichromats, while keeping chromatic components the same for more natural viewing by viewers with normal vision. Our method also falls in the category of optimization methods that aim to preserve original contrasts between colors.

Other recoloring methods take different approaches that do not require optimizing for new colors. Daltonize, online software by Dougherty and Wade [Doug], transfers red/green contrast to the lightness or blue/yellow axes, the axes protanopes and deuteranopes can see. Yang and Ro's method [Yang 03] adapts images for both dichromats and anomalous trichromats and is incorporated into the MPEG-21 framework [Song 03]. They adjust hue and saturation in HSV space for dichromats and perform a matrix multiplication in LMS space for anomalous trichromats. Meguro et al. [Megu 06] move colors along lines perpendicular to the confusion loci of a dichromatic viewer. Jefferson and Harvey [Jeff 07] create an interface for color deficient viewers

to interactively recolor images using a single slider. Huang et al. [Huan 07] rotate hue lines to map information from the a^* axis to the b^* axis in LAB space. In 2008, Huang et al. [Huan 08] remap hues according to a nonlinear hue transfer function calculated using generalized histogram equalization. Anagnostopoulos et al. [Anag 07], which is optimized for speed by Doliotis et al. [Doli 09], calculate the error between the original and simulated images and add this error into the original RGB image. Ma et al. [Ma 09] use a self-organizing map to learn a codebook of colors for an image which are then mapped to a better color palette for the viewer. There are a variety of ways to recolor an image for a color deficient viewer.

Although there are many methods that optimize images for color deficient viewers, these methods have different limitations that our method is designed to overcome. Some of the previous methods do not account for gamut and could produce solutions outside the target gamut or might not find solutions within the target gamut [Yang 03, Rasc 05a, Megu 06, Anag 07]. Other methods are designed to recolor webpages or documents and will not work for complex, natural images [Ichi 03, Waki 05]. A few methods require user input [Waki 05, Jeff 07]. There are methods that successfully increase contrast for the viewer but could produce images that look unnatural to the viewer [Doug, Rasc 05b, Anag 07, Ma 09]. Some methods find solutions by searching only in one dimension, such as hue [Huan 07, Huan 08], lightness [Tana 10], or along a normal direction [Megu 06]. In contrast, our method has the flexibility to find solutions within the entire target space while respecting the bounds of the target gamut. It is designed to work for natural images, and it strives to produce a result that appears natural to the viewer. Moreover, our method is unique because we show that it can be generalized to work for other problems, including color to gray, gamut mapping, and multispectral image fusion.

3.3.4 Products

Some products exist to aid color deficient viewers in everyday life. ChromaGen is a contact lens that helps color deficient viewers better discriminate between colors. The lens is a colored filter that is put in one eye. The filters come in nine colors and three intensities to support different types of color deficiencies. This technology, originally developed by David Harris [Harr 97], is available through Cantor and Nissel as contact lenses or glasses. More information about ChromaGen can be found in a practitioner's report [Hodd 98]. Additionally, Daniel Flück posted a list of twenty iPhone apps on his webpage on color blindness [Fluc 10]. These apps per-

form tasks such as provide the name of a color, simulate color deficient vision, daltonize an image, find colors that match a given color, and find colors that harmonize well together. Also, there is an extension to Google Chrome called Chrome Daltonize! [Goog] which can either simulate dichromatic vision or enhance images for protanopes, deuteranopes, or tritanopes.

3.4 Multispectral Image Fusion

Multispectral image fusion is a broad field with many methods and applications. The encompassing goal of these methods is to combine several spectral images into a single fused image that contains more information than any one spectral image alone. There is more published work in this domain than can be covered within this chapter, and there are surveys that give more details including recent surveys by Smith and Heather [Smit 05] and Goshtasby and Nikolov [Gosh 07]. Common method type categories include false color band replacement, IHS-based, PCA-based, pyramidal, wavelet-based, region-based, adaptive, and gradient-based. Additionally, there has been significant work on combining visible and infrared images. Alternatively, optical approaches have been developed for filtering light before it reaches the sensor or eye.

3.4.1 False Color Band Replacement

A popular way of visualizing spectral images is to replace each of the R, G, and B channels with an image from a single spectral band or principal component. Lillesand et al. [Lill 08] refer to a study which revealed interpreter preferences for band combinations in the RGB channels. A "normal color" image replaces the R, G, and B channels with bands 3 (red), 2 (green), and 1 (blue) of multispectral images taken from the Landsat 7 satellite. The "color infrared" composite replaces R, G, and B with bands 4 (near infrared), 3 (red), and 2 (green). The preferred false color images pair visible bands with infrared bands, enhancing vegetation discrimination since vegetation is highly reflective of infrared. Principal component analysis (PCA) [Joll 02] is also a common way to reduce high spectral dimensionality to three principal components containing the most variation. These three components can then be displayed in the R, G, and B channels to create a false color image, enhancing visualization of global variation in the multispectral image. In contrast, our method aims to display local variation in a natural color image close to the "normal color" image.

3.4.2 IHS, PCA, Pyramidal, Wavelet, and Region-Based Methods

Methods for multispectral fusion can be categorized by algorithmic approach. The most widely used approaches are based on Intensity-Hue-Saturation (IHS), Principal Component Analysis (PCA), pyramids, and wavelets. IHS methods were developed for merging low resolution multispectral images with high resolution panchromatic images to get a high resolution multispectral image. Panchromatic images are grayscale images captured by a sensor that is sensitive to all wavelengths of visible light and possibly infrared. The multispectral image contains the color information while the panchromatic image contains the spatial information, so merging the two would yield a high spatial resolution image with the colors of the multispectral image. IHS methods transform the multispectral image to IHS (Intensity, Hue, Saturation) space, replace the Intensity component with a modified panchromatic image, and apply an inverse transform [Carp 90, Shet 92, Choi 06]. PCA methods extract principal components, orthogonal axes with highest variation, from the multispectral data [Chav 89, Tyo 03, Lill 08, Cui 09]. Pyramid methods decompose source images into pyramids, construct a composite pyramid containing all the salient information, and invert the decomposition to get the output image [Toet 89, Toet 90, Toet 92, Burt 93, Wils 97]. Wavelet methods apply a wavelet transform to source images and modify wavelet coefficients [Li 95, Chip 95, Kore 95, Yock 95, Yock 96, Rock 97, Nune 99, Niko 01, Hill 05]. Other methods use high pass filtering [Chav 86], projection pursuit to find the optimal projection [Hari 96], or self organizing maps for a nonlinear mapping [Mand 96]. Most fusion methods operate at the pixel level, but some region-based methods exist [Zhan 97, Piel 03, Lewi 07, Miti 07]. These methods segment images into regions or features and fuse at that level. Also, there are methods that adapt according to input [Niko 07, Petr 07] or incorporate user constraints [Lawr 10]. All of these methods combine multiple images and have the same goal of enhancing visualization of information in the fused output image. Our method is another way to perform multispectral image fusion with the goal of preserving contrasts in a natural way for aesthetic visualization. In addition to preserving contrasts lost during the dimensionality reduction, our method also supports transformations from a source space to an arbitrarily constrained target space, such as gamut mapping, where the information loss comes from the constraints of the target space.

3.4.3 Gradient-Based Methods

Gradient methods based on Socolinsky and Wolff's method [Soco 02] convert multispectral images to grayscale. Socolinsky and Wolff compute structure tensors to get the gradient at each pixel. The gradient field, representing local contrast, is used to Poisson solve for the grayscale image that preserves local contrast. Similarly, Piella [Piel 09] uses structure tensors and solves for the image that has similar tensors by minimizing an objective function. Alsam and Drew [Alsa 09a] speed up Socolinsky and Wolff's method by replacing the eigenvalue computation with a max gradient selection. These methods output a grayscale image that best matches the desired gradients, preserving local contrast.

3.4.4 Visible + Near-Infrared Fusion

Recently, there have been many papers demonstrating the benefit of combining visible images with infrared (IR) images. The goal of these methods is to enhance visible images with IR information for improved contrast or detail visibility. Bennett et al. [Benn 07] enhance the underexposed regions of RGB video with details from the simultaneously recorded near infrared video. Zhang et al. [Zhan 08] combine RGB and near infrared (NIR) images of the same scene, fusing the contrast and texture of the NIR image into the visible image. Krishnan and Fergus [Kris 09] introduce dark flash photography where an NIR and UV (ultraviolet) flash replaces a normal visible light flash. This allows them to capture details of a low light scene without an intrusive flash. A second no-flash RGB image is taken to capture the ambient lighting. The two images are then fused, yielding a better image than could be captured in low light conditions without a visible flash. Süsstrunk and Fredembach [Suss 10] discuss methods for fusing visible and near-infrared images for a variety of applications. Other applications of combining visible and infrared images include enhancing night imagery [Fay 00, Das 00, Toet 02, Tsag 05] or aiding face recognition [Hari 06, Sing 08]. These successful applications prove that combining information from both visible and infrared images can produce an image that surpasses either image alone.

3.4.5 Optical Filtering

Optical filtering techniques modulate light before it reaches a sensor or a human eye. Mohan et al. [Moha 08] develop an optical approach that can be viewed as a spectral dimensionality reduction technique. Their Agile

Spectrum Imaging system is a prototype for a camera or projector with spectral filtering capabilities. They insert an LCD (liquid crystal display) into the optical path as a spectral filter, controlling the color spectra captured/projected. The LCD is placed in the "rainbow plane" created by a diffraction grating. By controlling the LCD values, they can selectively filter the spectral bands, reducing the spectral bands seen by the camera. In their prototype projector, they demonstrate how spectral filtering can be used to adapt color primaries to the image being projected, thereby adjusting the gamut for better color rendering for that image. In this way, spectral filtering can reduce spectral dimensionality to a few specially chosen color primaries. Wetzstein et al. [Wetz 10] develop SOPhIE, a see-through optical device that spatially filters light from a scene as the viewer looks through it, ultimately envisioned for applications such as glasses or car windshields. It consists of a spatial light modulation display, or LCD panel, inserted between the scene and the viewer and a camera that captures the unmodulated scene. By displaying different transmission images on the LCD panel, the light reaching the human eye is altered, and SOPhIE can be used for contrast reduction/enhancement, object highlighting, color satuation/de-saturation, de-metamerization, and visual enhancement for color deficient viewers.

3.5 Other Related Work

Other related work includes color quantization, tone mapping, dimensionality reduction, and colorization.

Other similar problems include color quantization and tone mapping. Color quantization maps a set of colors to a smaller set of colors in a way that best represents the original colors in the image [Heck 82]. When tone mapping a high dynamic range image to fit within a lower dynamic range, the goal is to preserve visual detail from the high dynamic range within the compressed space so that the output image looks like the input image [Tumb 93].

For general dimensionality reduction from n to m dimensions, there are several standard methods including Principal Component Analysis (PCA) [Joll 02] and Multidimensional Scaling (MDS) [Krus 78]. To reduce dimensionality with PCA, the data is projected onto the first m principal components. These principal components form a set of orthogonal axes in which each subsequent principal component lies along the direction of the largest remaining variance in the data. Important details in the other $n - m$ axes are lost. Furthermore, the resulting m dimensional values can be very different

from the results of the standard, natural projection, causing the output to look unrealistic. MDS reduces dimensionality by minimizing a stress function that makes output distances representative of original distances. Points that are close in the original space remain close in the output space. Rasche et al.'s color to gray method [Rasc 05b] is a case of MDS, and as they pointed out, MDS does not scale well to large datasets.

On the other end of the spectrum, colorization is a dimensionality expansion problem that adds color to a grayscale image [Wels 02, Levi 04, Iron 05]. Levin et al. [Levi 04] solve an optimization problem to propagate colored user scribbles through space without crossing intensity boundaries. A related problem of transferring color between two images is solved in Reinhard et al. [Rein 01] and Jia et al. [Jia 04].

Chapter 4

Cluster Based Contrast Improvement

We apply the same basic method to the applications of color to gray conversion, gamut mapping, image optimization for color deficient viewers, and multispectral image fusion. For all of these problems, our two main goals are to preserve contrasts and maintain naturalness. This chapter presents the general method of the framework while application-specific details are described in each application's chapter.

Our cluster-based method locally improves contrast in five steps. An overview of our method is depicted in the pipeline diagram in Figure 4.1. In the schematic diagram and in the rest of this chapter, we use images from color to gray conversion for illustration purposes, but the method generalizes to the other applications discussed in this thesis.

1) **Projection to target space.** We project the original image to the target space to get the initial mapped image whose contrast we aim to improve.

2) **Clustering.** We cluster pixels spectrally and spatially to get areas that might exhibit local contrast. We compute cluster statistics including mean, covariance, and approximate radius.

3) **Graph creation.** We create a graph connecting spatially close clusters, where graph edges represent the local contrasts between neighboring clusters.

4) **Optimization.** We solve for new cluster colors in the target space with a least squares optimization that aims to preserve original contrast.

5) **Blending.** Instead of applying the same translation to all pixels in a cluster, we calculate new pixel colors using a weighted blend of cluster translations.

To summarize, our cluster-based method groups pixels into clusters according to their spectral and spatial similarities, improves contrast between clusters by translating the clusters to optimal colors within the target space, and transfers those translations back to the pixels.

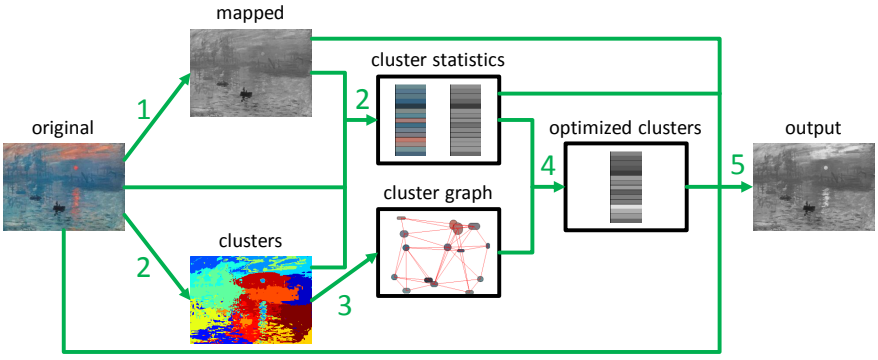


Figure 4.1: **Method overview.** 1) We project the original image to the target space to get the initial mapped image. 2) We cluster the pixels in spatio-chromatic space and compute cluster statistics. The cluster image as well as original and mapped cluster means are shown here. 3) We create a graph by connecting spatially close clusters. 4) We optimize for new cluster colors in the target space. 5) We transfer the optimization results back to the pixels with a blending step. *Original image courtesy of Gooch et al. [Gooc 05].*

Our method is formed around the concept of contrast. Colloquially, contrast is the difference between two stimuli that makes the stimuli distinguishable from each other. Many of the applications in this thesis have input and output data that can be expressed in a perceptually uniform color space such as CIE LAB, CIE LUV, or IPT. In these spaces, we can compare perceptual differences between colors by comparing the Euclidean distances between the colors. We use the Euclidean distance metric in a perceptually uniform space as our measure of contrast. For example, to compare the contrast between clusters before and after the initial mapping, we compare the Euclidean distance between the clusters before the mapping to the Euclidean distance between them after the mapping. The target contrast enhancement for a cluster pair is also specified in terms of Euclidean distance as it comes from the length of the target vector. Larger distances correspond to larger contrasts.

4.1 Projection to Target Space

One of our goals is to maintain naturalness, where what is "natural" is defined by some mapping that projects the input image from its source space $\mathcal{S}$ in $\mathbb{R}^n$ to its target space $\mathcal{T}$ in $\mathbb{R}^m$. Our optimization aims to combat the local contrast loss imposed by this initial mapping, while staying close to it. For each application, we assume a standard mapping from source space to target space exists. For example, we get an initial gamut mapped image by applying HPMINDE clipping to map the source image to the target gamut. To project color to gray, we use standard mappings such as luma or CIE L^*, but any projection from the source space to the target space could be used to get the initial mapped image.

In order to make distances perceptually comparable in the source space, we first losslessly transform the input data, where possible, to a perceptually uniform space such as CIE LUV. Where we have 3D color input data, this transformation yields better clustering results and target vector calculations in the optimization step.

4.2 Clustering

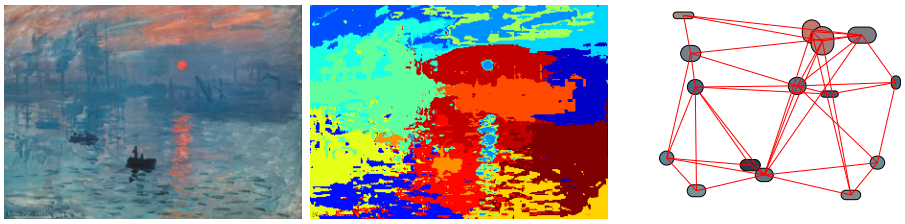
To separate the image into areas that might exhibit local contrast between each other, we cluster pixels in spatio-chromatic space $\mathcal{U}$, in $\mathbb{R}^{n+2}$, formed by adding spatial dimensions to source space dimensions. Each pixel has a feature vector $\mathbf{u} = (\mathbf{s}, x', y')$ where $\mathbf{s} \in \mathcal{S}$ and (x', y') are the (x, y) pixel coordinates multiplied by a spatial weight that defines the tradeoff between spatial and chromatic dimensions. We cluster the pixels using k-means++ [Arth 07], which selects initial seeds intelligently with a higher probability of picking points that are farther from the seeds that are already picked. The number of clusters is a user defined parameter that should be chosen so that the source space cluster means are representative colors of constituent pixels but also so that resulting clusters roughly resemble image features. We iterate until the pixel assignments stop changing or until a maximum number of 50 iterations is reached. Figure 4.2(b) shows an example output of the clustering step.

In order to preserve and account for intra-cluster detail, we calculate cluster means and covariance matrices, defining ellipsoids, which are used later in the optimization and blending steps. Each cluster's mean serves as the representative color for that cluster during the optimization step. To make sure the covariance matrices are invertible we enforce a minimum

ellipsoid radius by clamping small eigenvalues to a minimum value of 10^{-6}. For simplicity, each cluster is also approximated by a sphere with radius r equal to the average length of the cluster's principal components. Although k-means clustering is run only on data in $\mathbb{R}^{n+2}$, we use the resulting clusters to also compute statistics for the original clusters in the source space in $\mathbb{R}^n$ and the mapped clusters in the target space in $\mathbb{R}^m$.

4.2.1 Spatial Weight

The spatial weight is based on the number of JNDs (just noticeable difference units) per length subtended by a 2° visual field, and it takes into account image size, viewing distance, and monitor resolution. We let the number of JNDs per length subtended by a 2° visual field be equivalent to $l = 2$. For a $w_M \times h_M$ monitor with resolution ρ viewed at distance v, the number of pixels per JND is $\frac{2v\rho\tan(1°)}{l}$. To make clustering results independent of image size, instead of finding how many pixels correspond to one JND, we find the percentage of the image size that corresponds to one JND. For this we need a linear metric of image size such as the square root of the number of pixels in the image. The percentage of image size that should equate to one JND is then $\frac{2v\rho\tan(1°)}{l\sqrt{w_M h_M}}$. The spatial weight that converts (x, y) pixel coordinates to equivalent values in the source space is $\frac{lb\sqrt{w_M h_M}}{2v\rho\tan(1°)\sqrt{w_I h_I}}$ where b is the number of units in the source space equivalent to one JND.



(a) original (b) clusters (c) cluster graph

Figure 4.2: **Clusters.** Pixels of the original image are clustered spectrally and spatially into 15 clusters shown color coded in (b). A graph (c) is created in which each cluster is a vertex. Edges connect spatially close clusters and represent the local contrasts we want to preserve. *Original image courtesy of Gooch et al. [Gooc 05].*

4.3 Graph Creation

We represent areas of local contrast by creating a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ connecting spatially close clusters. Graph edges represent local contrasts between clusters or, more specifically, the relative original contrasts our optimization aims to preserve. Since we focus on enhancing local contrast over global contrast, we only consider contrast between spatially close clusters and exclude spatially distant pairs to avoid adding extraneous constraints to the optimization.

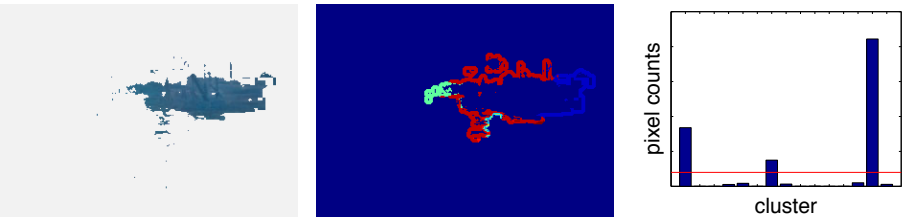
Graph edges are created based solely upon the clusters' spatial extents shown in the visualization in Figure 4.2(b). An example cluster mask depicting a cluster's spatial extent is shown in Figure 4.3(a). Since we only want to consider the main bulk of a cluster's spatial extent when determining spatial neighbors, we first remove outliers from each cluster's mask. Our outlier removal strategy dilates the cluster mask and rejects disconnected regions containing less than 5% of the cluster's pixels. The disk dilation radius starts at $\delta = 5$, which is sufficient for almost all clusters, and increases in multiples of 5 until at least 80% of cluster pixels are valid. This gives us a valid cluster mask which is used during the rest of the graph creation step.

Once outliers are removed, graph edges are created by examining the dilated region around a cluster and counting pixels from potential neighbors. We use dilation again (with disk radius $\delta = 5$), but this time we dilate the valid cluster mask to see which neighbors are in the dilated region in Figure 4.3(b) and, therefore, spatially touching. Then, the pixels in the dilated region are counted to produce the histogram of potential neighbors in Figure 4.3(c). For each cluster, we count the pixels in its dilated region belonging to other clusters, resulting in a histogram per cluster. Clusters i and j are neighbors and (i, j) is an edge in $\mathcal{E}$ if both clusters have large counts above a threshold in each other's histogram. Figure 4.2(c) shows the resulting cluster graph. We can optionally consider more spatially distant pairs by adding edges to the two-neighbors or three-neighbors from the original graph.

4.3.1 Thresholds

When deciding whether histogram counts are large enough, two types of thresholds are used, and either is accepted. The pixel threshold is usually set to 200 pixels. In cases where the input image is small, some neighboring clusters will not have 200 adjacent pixels. The pixel threshold is then set to be 5% of the number of image pixels divided by the number of clusters if this

number is smaller than 200. A percentage threshold, used to account for small clusters, accepts counts if they are above 10% of the smaller cluster's size.



(a) cluster mask (b) dilated region (c) neighbor counts

Figure 4.3: **Cluster graph creation.** (a) Pixels from a single cluster. (b) Dilated region, color coded by neighbor. (c) Histogram of pixel counts in dilated region. Clusters are neighbors if both clusters have large counts above a threshold (red) in each other's histogram.

4.4 Optimization

4.4.1 Linear Least Squares Optimization

Given the graph, we apply an optimization procedure to translate the clusters within the target space such that contrast is improved between clusters that have lost contrast. At the same time, we would like to remain close to the initial projection to the target space to preserve the naturalness of the image. We solve a linear least squares problem for the optimal cluster colors $\mathbf{x}$ that minimize

$$\mathbf{x} = \arg\min_{\mathbf{x}} E_T + \lambda_M E_M + \lambda_H E_H + \lambda_L E_L + \lambda_P E_P \tag{4.1}$$

where E_T is the term that preserves contrast, E_M is a regularization term that stays close to the initial mapping, E_H is the term that prevents hue shifts, E_L is the term that preserves the neutrality of achromatic colors, E_P is the term used in constrained optimizations to stay close to the previous iteration's output, and λ_M, λ_H, λ_L, and λ_P are parameter weights on their corresponding terms. Not all of the terms here are applicable to all transformation problems; only the first two terms are required. Since the goals of the framework are to simultaneously preserve contrast and remain close to

the initial mapping, all applications use the target and regularization terms, E_T and E_M. For some applications it can be beneficial to prevent artifacts such as hue shifts or chromatic shifts for achromatic colors. In these cases, the E_H or E_L terms could be included in the minimization function. For applications such as gamut mapping, our constrained optimization procedure, described in Section 4.4.2, solves Equation 4.1 iteratively and includes the E_P term to stay close to the previous iteration's output.

Target Term

The term E_T preserves contrast by matching cluster difference vectors to target vectors $\mathbf{t}_{ij}$ for all edges (i, j) in $\mathcal{E}$. The target term is

$$E_T = \sum_{(i,j)\in\mathcal{E}} \tau_{ij}((\mathbf{x}_j - \mathbf{x}_i) - \mathbf{t}_{ij})^2, \tag{4.2}$$

where τ_{ij} is a weight on edge (i, j). To increase the contrast between a cluster pair, we would like to lengthen its initial vector $\mathbf{m}_{ij}$ between the mapped clusters by an amount based on the visible contrast lost. Therefore, we set the target vector $\mathbf{t}_{ij}$ to the initial mapped vector $\mathbf{m}_{ij}$ and let its magnitude be the initial magnitude m_{ij} lengthened by some additional magnitude a_{ij}, as defined below.

$$\begin{aligned}
\mathbf{t}_{ij} &= \frac{m_{ij} + a_{ij}}{m_{ij}}\mathbf{m}_{ij} \\
a_{ij} &= k \cdot S(\psi_{ij}(o_{ij} - \left\|F(\mathbf{m}_j) - F(\mathbf{m}_i)\right\|)) \\
\psi_{ij} &= \frac{1}{1 + e^{-\kappa(\left\|F(\mathbf{m}_j) - F(\mathbf{m}_i)\right\| - c)}} \\
S(x_{ij}) &= \frac{\max(0, x_{ij})}{\max_{(i,j)\in\mathcal{E}} x_{ij}}
\end{aligned} \tag{4.3}$$

Cluster pairs that lose a lot of contrast, such as those with different source colors that map to the same target color, need their contrast restored the most and thus receive the largest additional magnitudes up to the user defined parameter k. The additional magnitude a_{ij} is based on how far apart clusters are before and after the initial mapping. Direct comparisons of distances in the source space to distances in the target space are not ideal for all cases, especially when a transformation exists that will bring the target space and source space clusters into a common color space. For a_{ij}, the L_2 distance o_{ij} between the original clusters in the source space is compared to $\left\|F(\mathbf{m}_j) - F(\mathbf{m}_i)\right\|$, the comparable L_2 distance between the

mapped clusters in the modified target space. The function $F()$, which varies for each application, transforms the target space clusters $\mathbf{m}$ in order to make the distances from the target space comparable to the distances in the source space. Since we do not need to enhance contrasts that are already visible, we multiply by a sigmoidal weight ψ_{ij}, giving those cluster pairs less additional magnitude. The sigmoid is fairly steep with $\kappa = 80$ centered around a distance in the target space determined from our image data to indicate when clusters already have sufficient visible contrast. The $S()$ function clamps negatives to zero and scales values so that the max for all pairs $(i, j) \in \mathcal{E}$ equals one. This results in final a_{ij} values in $[0, k]$.

Critical edges are the cluster pairs that most need contrast enhancement. Edge (i, j) is critical if the initial mapping causes clusters i and j to overlap more in the target space than they did in the source space. Since it is more important for critical edges to be enhanced, the per edge weight τ_{ij} gives more weight to edges that are closer to critical edges in the graph, allowing these edges to better match their targets. The weight τ_{ij} is defined as

$$\tau_{ij} = \begin{cases} \frac{1}{|\mathcal{E}|}, & \sum_{(i,j)\in\mathcal{E}} e^{-g_{ij}} = 0, \\ \frac{e^{-g_{ij}}}{\sum_{(i,j)\in\mathcal{E}} e^{-g_{ij}}}, & \sum_{(i,j)\in\mathcal{E}} e^{-g_{ij}} > 0, \end{cases} \tag{4.4}$$

where g_{ij} is the shortest path length from edge (i, j) to the nearest critical edge in $\mathcal{G}$, and $|\mathcal{E}|$ is the number of edges in $\mathcal{E}$. In this case, path length is the smallest number of nodes in $\mathcal{G}$ connecting edge (i, j) to the closest critical edge. The denominator normalizes the per edge weights so that they sum to one, i.e., $\sum_{(i,j)\in\mathcal{E}} \tau_{ij} = 1$. If no critical edges exist in the graph, then path lengths will be ∞, $e^{-\infty} = 0$, and every edge receives equal weight.

If the source space and the target space have the same dimensions, i.e. $\mathbb{R}^n = \mathbb{R}^m$, then we can easily set the target vectors to the original source vectors $\mathbf{o}_{ij}$ between clusters in lieu of Equation 4.3. This preserves the original vector directions and magnitudes, which we cannot do when the source and target spaces have different dimensions.

Regularization Term

The term E_M keeps the output cluster colors close to the initial mapping $\mathbf{m}$ to the target space, thereby maintaining the appearance of the standard mapping. The regularization term is

$$E_M = \sum_{i\in\mathcal{V}} \tau_i (\mathbf{x}_i - \mathbf{m}_i)^2, \tag{4.5}$$

where the per cluster weight τ_i gives areas with critical clusters more freedom to move by giving less weight to clusters that are closer to critical clusters in $\mathcal{G}$. A cluster is critical if it is part of a critical edge, an edge for which contrast enhancement is the most desired. The per cluster weight τ_i helps critical clusters achieve their target contrast but also helps noncritical clusters remain closer to their natural colors. The weight τ_i is defined as

$$\tau_i = \frac{1 - e^{-(g_i+0.1)}}{\sum_{i\in\mathcal{V}} 1 - e^{-(g_i+0.1)}}, \tag{4.6}$$

where g_i is the shortest path length from cluster i to the nearest critical cluster in $\mathcal{G}$. The constant 0.1 is used to avoid zero weights on critical clusters when $g_i = 0$. As with the per edge weights, the per cluster weights are also normalized to sum to one, i.e. $\sum_{i\in\mathcal{V}} \tau_i = 1$.

Hue Term

For some applications, such as gamut mapping, where it is important to preserve hue, we include the term E_H in Equation 4.1 weighted by parameter λ_H. This prevents hue shift by penalizing cluster movement in the direction orthogonal to each cluster's plane of constant hue. The matrix H is a block diagonal matrix composed of blocks $H_1, \ldots, H_N$ where N is the number of clusters. Each block H_i is an $m \times m$ matrix that projects the optimized color $\mathbf{x}_i$ in $\mathbb{R}^m$ onto the space orthogonal to cluster i's constant hue plane. The hue term is

$$E_H = \big\|H\mathbf{x}\big\|^2. \tag{4.7}$$

Achromaticity Term

Where applicable, we preserve achromaticity, ensuring that neutral source colors remain neutral in the target space by including the term E_L in Equation 4.1 weighted by parameter λ_L. For achromatic source colors, the term penalizes cluster movement off the neutral axis of the target space. In the block diagonal matrix Q, structured similar to H, each block Q_i projects the cluster movement vector $\mathbf{x}_i - \mathbf{m}_i$ onto the subspace orthogonal to the neutral axis. In the matrix $L = WQ$, these projections are weighted by the achromaticity weights in the diagonal matrix W. For each cluster, the achromaticity weight is a Gaussian function of the original cluster's chroma value with $\sigma = 10$ JND units. Each weight is repeated m times on the diagonal of W. The achromaticity term is

$$E_L = \big\|L(\mathbf{x} - \mathbf{m})\big\|^2. \tag{4.8}$$

Critical Pairs

Critical pairs are pairs of clusters that most need contrast enhancement such as those that are far away from each other in the source space but map to similar locations in the target space. We categorize an edge between a cluster pair as critical when the initial mapping has caused the clusters to overlap more in the target space than they did in the source space. We rely on a simplified, spherical approximation of clusters to calculate the amount of overlap for a cluster pair. The overlap amount υ for cluster pair (i, j) is defined as

$$\begin{aligned} \upsilon_{o_{ij}} &= o_{ij} - r_{o_i} - r_{o_j} \\ \upsilon_{m_{ij}} &= m_{ij} - r_{m_i} - r_{m_j} \end{aligned} \tag{4.9}$$

where $\upsilon_{o_{ij}}$ is the overlap or separation between the original clusters in the source space, and $\upsilon_{m_{ij}}$ is the overlap or separation between the mapped clusters in the target space. The Euclidean distances between the original and mapped cluster means are o_{ij} and m_{ij}, respectively. The overlap or separation amount is the distance between the means less the radii of clusters i and j. The original clusters in the source space have radii r_o, and the mapped clusters in the target space have radii r_m. Figure 4.4 shows the three scenarios of cluster overlap. Positive overlap values mean the clusters do not touch, zero means the clusters touch, and negative means they overlap. Overlap amounts before the mapping use distances and radii from the original clusters in the source space while overlap amounts after the mapping use distances and radii from the mapped clusters in the target space. This means the overlap amounts are in $\mathbb{R}^n$ and $\mathbb{R}^m$, so in order to compare the overlap before and after the mapping, we compare overlap amounts as fractions of cluster radii. An edge is critical if the clusters touch after the mapping and if, for both clusters, their fractional overlap is greater after the mapping than before the mapping. More formally, edge (i, j) is critical if

$$\left(\frac{N(\upsilon_{m_{ij}})}{r_{m_i}} > \frac{N(\upsilon_{o_{ij}})}{r_{o_i}}\right) \,\&\, \left(\frac{N(\upsilon_{m_{ij}})}{r_{m_j}} > \frac{N(\upsilon_{o_{ij}})}{r_{o_j}}\right), \tag{4.10}$$

where $N(x) = |\min(x, 0)|$.

The per edge and per cluster weights, τ_{ij} and τ_i, are assigned based on the edge's or cluster's proximity to critical edges or clusters. We use path length in $\mathcal{G}$ as a measure of distance to a critical edge or cluster. The critical edges and clusters themselves have distance values of zero. The distance for a noncritical cluster i would be the minimum number of edges between i

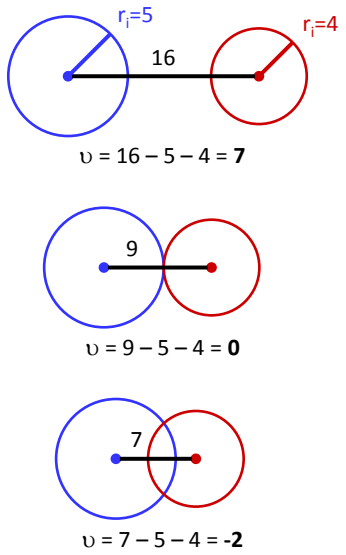


Figure 4.4: **Amounts of overlap for cluster pairs.** Two clusters in a pair either do not touch (resulting in a positive overlap amount), touch (resulting in an overlap amount of zero), or overlap (resulting in a negative overlap amount). Overlap amounts are used to find critical pairs, cluster pairs that overlap more in the target space than they did in the source space.

and its nearest critical cluster in $\mathcal{G}$. The distance for a noncritical edge (i, j) would be the minimum number of clusters between (i, j) and its nearest critical edge in $\mathcal{G}$.

Solving the Optimization

We solve for the optimal cluster colors $\mathbf{x}$ that minimize Equation 4.1, possibly modified to include the hue-preserving or achromaticity-preserving terms in Equations 4.7 and 4.8 respectively. We use Matlab's `lsqlin()` method to find the least squares solution. For target spaces with axis aligned boundaries in $\mathbb{R}^m$, we can constrain each cluster mean to stay within r_{m_i} away from the target space boundary where r_{m_i} is the mapped cluster's radius. This can be done with simple linear constraints on each cluster. The linear least squares problem with linear constraints is convex and can be solved to give us the globally optimal solution. However, when the target space is not nicely axis aligned, we resort to our constrained optimization procedure to

solve for cluster colors within arbitrary shaped target spaces.

4.4.2 Constrained Optimization

For applications such as gamut mapping, we need to constrain the optimized cluster colors to lie within a finite gamut $\mathcal{T}$ in $\mathbb{R}^m$ where $\mathcal{T}$ cannot be described by simple bound constraints on $\mathbb{R}^m$. In this case, we perform a constrained optimization in which we solve the linear least squares problem for the optimal cluster colors in $\mathbb{R}^m$, project the optimized results to the target gamut $\mathcal{T}$, and iterate. We incorporate each previous iteration's result by including the term E_P weighted by parameter λ_P in our least squares problem in Equation 4.1, causing our optimization to converge to a solution within the arbitrary shaped target space $\mathcal{T}$. On iteration z, we solve for the optimized cluster colors $\mathbf{x}^{(z)}$ that minimize

$$\mathbf{x}^{(z)} = \arg\min_{\mathbf{x}^{(z)}} E_T + \lambda_M E_M + \lambda_P E_P + \lambda_H E_H + \lambda_L E_L, \qquad z = 0, \ldots, Z-1 \tag{4.11}$$

where the hue and achromaticity preserving terms from Equations 4.7 and 4.8 have been included. Then, we project the optimized colors $\mathbf{x}^{(z)}$ to the target space with projection operator $P()$, yielding projected output colors $\mathbf{p}^{(z)} = P(\mathbf{x}^{(z)})$. This iterative procedure is not guaranteed to find a globally optimal solution but rather converges to a local minimum near the standard mapping.

Previous Iteration Term

The term E_P keeps the optimized cluster colors $\mathbf{x}^{(z)}$ close to the projected output $\mathbf{p}^{(z-1)}$ of the previous iteration $(z-1)$. On the first iteration $(z=0)$, we exclude the term E_P from Equation 4.11 by letting $E_P = 0$. The previous iteration term is

$$E_P = \begin{cases} 0, & z = 0, \\ \sum_{i \in \mathcal{V}} (\mathbf{x}_i^{(z)} - \mathbf{p}_i^{(z-1)})^2, & z = 1, \ldots, Z-1. \end{cases} \tag{4.12}$$

4.4.3 Parameter Settings

The parameters in our optimization are set to constants with the exception of k and possibly λ_M. We found that $\lambda_P = 0.5$, $\lambda_H = 4$, $\lambda_L = 10$ works well against an implied unity weight on the target term. The weights on the hue and achromaticity preserving terms are set very high because artifacts due to hue shifts or chromaticity gains are known to be particularly disturbing

in the gamut mapping literature. We found that setting $\lambda_M = 0.8$ for the regularization term works well to preserve the realistic nature of the scene, but we can decrease λ_M if we want to allow the optimization to deviate further from the natural mapping. The parameter k, as the max magnitude increase for target vectors, controls the amount of contrast enhancement. This is the only true variable in our approach as it controls the application-specific tradeoff between staying close to the generic mapping and recovering the lost contrast from the source space. In our examples, we typically set k in $[0.1, 1]$ depending on how much contrast we want to add to the image. Section 9.2 talks more about the effect of setting these parameters to different values.

4.5 Blending

Every pixel in a cluster is not exactly represented by its cluster's mean. In fact, a pixel given by its vector $\mathbf{u} \in \mathcal{U}$ could be located between the means of neighboring clusters in $\mathbb{R}^{n+2}$, and for these pixels it is better to blend between the results of the neighboring clusters. The optimization step solves for the output cluster colors $\mathbf{x}$ in the target space (or for the projected output $\mathbf{p}$ in the constrained optimization case). Using the mapped cluster colors $\mathbf{m}$ we find difference vectors $\mathbf{d}_i = \mathbf{x}_i - \mathbf{m}_i$ for each cluster i. Instead of applying the same difference vector to every pixel in the cluster, a difference vector $\mathbf{d}_q$ for each pixel q is calculated as a weighted combination of difference vectors over all clusters.

$$\mathbf{d}_q = \sum_{i \in \mathcal{V}} \omega_{qi} \mathbf{d}_i \tag{4.13}$$

$$\omega_{qi} = \frac{1}{(\mathbf{u} - \mu_i)^T M_i \mathbf{u}}$$

For each pixel, the weight ω_{qi} on each cluster's difference vector is inversely proportional to the pixel's squared Mahalanobis distance to cluster i. The squared Mahalanobis distance is $\mathbf{u}$ evaluated in the cluster's ellipsoid equation defined by cluster mean μ and inverse covariance matrix M. Near zero weights are clamped to $1e^{-6}$, and the weights are normalized to sum to one for each pixel.

Each pixel is updated by adding its difference vector $\mathbf{d}_q$ to its mapped color vector. In applications, such as gamut mapping, where the source and target spaces are both in $\mathbb{R}^n$, we can use source clusters and pixels to compute difference vectors and update pixel values, preserving intra-cluster detail. By blending the optimization results, we smooth the output

where necessary and avoid quantization-like discontinuities. Artifacts from misclassifying pixels in the clustering step are also subdued since those pixels will get higher weights for the clusters to which they are actually closer.

Finally, to ensure that all colors are within $\mathcal{T}$, the output pixel colors are clipped to the target space.

Chapter 5

Color to Gray

5.1 Introduction

When converting color images to grayscale, chromatic contrast is lost. For example, the contrast between the red sun and the blue background in Figure 5.1(a) is lost in its grayscale counterpart in Figure 5.1(b). Even though the red and blue colors of the original image are strikingly different, the sun is barely visible in the grayscale version. This contrast loss is the main problem that recent color to gray algorithms aim to solve.

(a) color (b) standard gray (CIE L^*)

Figure 5.1: **Contrast lost.** The strong color contrast between the red and blue elements of Monet's painting is lost upon conversion to grayscale using a standard projection such as CIE L^*. *Original image courtesy of Gooch et al. [Gooc 05].*

Color to gray conversion is an important transformation because it is frequently used when sending color images to monochrome devices such as printers. Printing in grayscale is still a common task today. When printing color images, printers need to map colors to grayscale, and each device can have its own transformation function. For example, some standard projections to grayscale include luma or CIE L^*. The luma conversion takes a weighted sum of nonlinear sRGB color channels. The weights could be specified by the commonly used Rec. 709 [Recob] or Rec. 601 [Recoa] coefficients, for example, or they could be specially chosen and built into a

device. Regardless of the specifics of the coefficients, the weighted combination reduces three channels of information to one. With L^*, colors in 3D space are projected to the 1D L^* axis, as depicted in Figure 5.2. With any of these 3D to 1D projections, chromatic details are inevitably lost. Contrasts in the color image can be reduced to smaller grayscale contrasts that do not fully represent their color contrast counterparts. In the extreme case, isoluminant colors are mapped to the same graylevel, and the contrast between the colors is lost completely. Figure 5.2 uses CIE LUV space to illustrate what it means for colors to be isoluminant or have the same gray value; the entire plane of colors perpendicular to the L^* axis is mapped to a single gray value on the L^* axis.

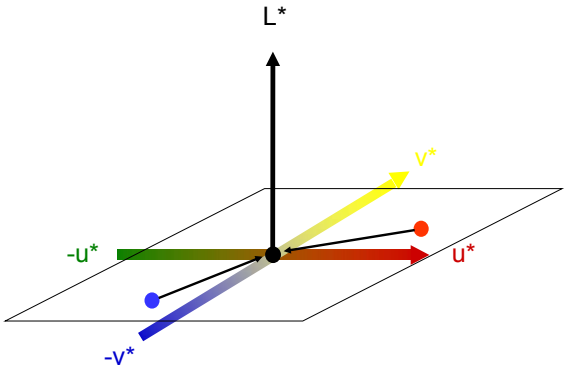
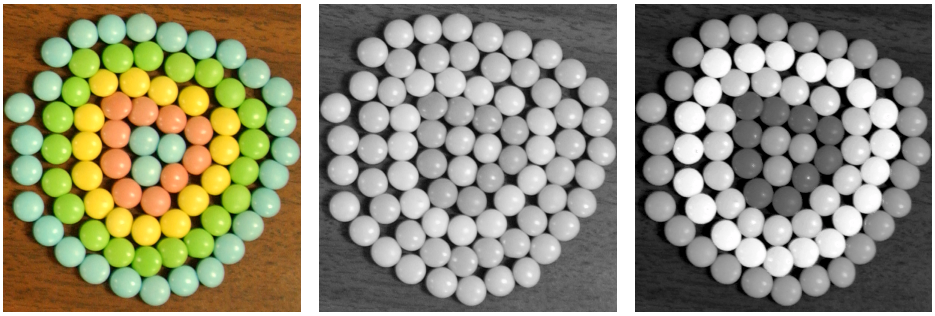


Figure 5.2: **Isoluminant colors.** During the grayscale projection to the L^* axis, colors in CIE LUV space are projected to their L^* values. All colors within a plane perpendicular to the L^* axis have the same L^* value. These colors are isoluminant colors because they are projected to the same gray value on the L^* axis.

The color to gray problem fits within our framework for problems suffering from information loss. Within the context of the framework, where n dimensions are reduced to m dimensions, $n = 3$ and $m = 1$ since a tristimulus color image is converted to a single dimension grayscale image. Similar to the other problems in our framework, this loss in dimensionality leads to metamerism, where isoluminant colors map to the same gray value.

We apply our method, outlined in Chapter 4, to the color to gray problem and develop a new color to gray method that realistically preserves chromatic details. Our first goal is to preserve the chromatic contrasts from the original

source image as changes in grayscale. In many images, color serves as an important visual cue that helps discern image features. For example, in Figure 5.1, the red sun against the blue background is the focal point of the image which is lost upon conversion to grayscale. We want to express changes in color as changes in grayscale because preservation of salient scene features is more important to human viewers than capturing accurate light intensities [Gooc 05]. On the other hand, if the output graylevels stray too far away from the standard gray values, the viewer will have trouble believing that the output could realistically come from the color image. For example, if to enhance the contrast between the rings of candy in Figure 5.3(a) the green ring of candy were given a gray value close to white, as in the global mapping in Figure 5.3(c), the resulting image would not be a credible grayscale version of the original color image. Therefore, our second goal is to keep the output gray values close to the standard grayscale projection so that the output is a plausible grayscale conversion of the color image. These two goals fit with the goals of our framework where we want to preserve the contrasts from the source space while remaining faithful to the standard mapping that defines a target appearance for the output.



(a) color (b) standard gray (luma) (c) Rasche et al. [Rasc 05b]

Figure 5.3: **Staying close to the standard for a natural result.** The grayscale output of Rasche et al. [Rasc 05b] in (c) preserves contrast between the rings but is not a believable grayscale version of the original because the resulting grayscale values are very different from the standard luma projection (b). *Original image courtesy of Rasche et al. [Rasc 05b].*

Our color to gray results illustrate how we are able to preserve the contrasts from the input color image within the single dimension grayscale axis in a realistic manner. Additionally, a benefit of our semi-local method is

that it improves contrast at edges without introducing halo artifacts. As discussed in Chapter 3 on Related Work, some of the previous color to gray methods use local filtering techniques which can introduce halo artifacts. In contrast, our cluster-based method improves local contrast between neighboring areas without causing halos. This is because an area is represented by a single cluster, and the area is modified as a single entity. Pixels within a cluster are consistently assigned gray values based on the output gray value for the whole cluster instead of being assigned values based on neighboring pixels. Similar to the other problems of the framework, our color to gray results show an improvement in contrast from the standard projection without deviating too far from it, maintaining naturalness.

5.2 Method

We apply the general method described in Chapter 4 to the conversion of color to gray. In this section, we explain the method in terms of the color to gray problem and include application-specific details. First, we cluster together pixels with similar colors in the same spatial region in the image. Then, for the clusters, we solve for new gray values that reflect chromatic contrasts from the source space as differences in gray values. Finally, we transfer the new graylevels back to the pixels in a blending step.

5.2.1 Projection to Target Space

We first convert the input color image, usually represented in sRGB space, to a perceptually uniform space. This lossless transform allows us to compare Euclidean distances between colors as perceptual differences. We choose to use CIE LUV as the perceptually uniform source space $\mathcal{S}$.

Projecting the color image to a standard gray image gives us an initial gray image which serves as our target output appearance. Our optimization aims to improve upon the contrasts lost during the grayscale projection while staying close to the standard projection for a natural result. We project the LUV source colors from the source space $\mathcal{S}$ in $\mathbb{R}^3$ to the grayscale target space $\mathcal{T}$ in $\mathbb{R}^1$. For the target gray, we use standard projections such as luma (with Rec. 709 [Recob] coefficients) or CIE L^*, but other projections could be used such as CIE Y or the Helmholtz-Kohlrausch prediction used by Smith et al. [Smit 08]. Any projection from color to gray could be used to define the target appearance and, thus, influence the look of the output image.

5.2.2 Clustering

We cluster the image pixels in spatio-chromatic space $\mathcal{U}$ in $\mathbb{R}^5$ where each pixel's feature vector is (L, U, V, x', y'), the pixel's color dimensions augmented by its weighted spatial dimensions. This separates the image into local areas of similar colors. Each cluster represents an area, and the contrast between clusters represents the local color contrast between spatially neighboring areas. The weight on the spatial dimensions (x, y) balances the tradeoff between color and space as described in Section 4.2. Once the clusters are determined, we calculate statistics (i.e. mean, covariance matrix, and approximate radius) for three sets of clusters: the clusters in $\mathcal{U}$, the original color clusters in the LUV source space in $\mathbb{R}^3$, and the mapped grayscale clusters in $\mathbb{R}^1$. The means and standard deviations for the grayscale clusters are calculated from standard gray image pixels.

5.2.3 Graph Creation

We create the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of clusters, and the edges in $\mathcal{E}$ connect spatially close clusters. The edges are determined by the dilation-and-neighbor-counting procedure described in Section 4.3.

5.2.4 Optimization

The optimization solves for optimal gray values for the clusters that simultaneously improve contrasts between the initial gray clusters while remaining close to the initial gray means for a plausible grayscale result. We solve the linear least squares problem in Equations 4.1, 4.2, and 4.5, repeated here and simplified, for the optimal cluster gray values $\mathbf{x}$ that minimize the following.

$$\begin{aligned} \mathbf{x} &= \arg\min_{\mathbf{x}} E_T + \lambda_M E_M \\ E_T &= \sum_{(i,j)\in\mathcal{E}} \tau_{ij}((\mathbf{x}_j - \mathbf{x}_i) - \mathbf{t}_{ij})^2 \\ E_M &= \sum_{i\in\mathcal{V}} \tau_i(\mathbf{x}_i - \mathbf{m}_i)^2 \end{aligned} \tag{5.1}$$

Only the first two terms from Equation 4.1, shown above, are necessary for color to gray conversion, so we set the additional terms to zero. Since our target space is a one dimensional grayscale space, preserving hue and achromaticity does not make sense in this case. E_T is the term that preserves

contrast by matching target differences and E_M is the regularization term that stays close to the initial gray means from the standard gray projection. The weights λ_M, τ_{ij}, and τ_i are described in Section 4.4.

The target vectors are designed to enhance contrasts between clusters. They are described in Section 4.4 and defined as $\mathbf{t}_{ij}$ for each edge (i, j) in Equation 4.3, which is repeated below in Equations 5.2, 5.3, and 5.5.

$$\mathbf{t}_{ij} = \frac{m_{ij} + a_{ij}}{m_{ij}} \mathbf{m}_{ij} \tag{5.2}$$

Since the target gray space is in $\mathbb{R}^1$, the target vectors are actually 1D scalar differences in this space. The target difference $\mathbf{t}_{ij}$ for a cluster pair is the pair's initial gray difference $\mathbf{m}_{ij}$ increased by an additional amount a_{ij}. The sign of the target difference remains the same as the sign of the initial gray difference. The additional amount a_{ij}, defined as

$$a_{ij} = k \cdot S(\psi_{ij}(o_{ij} - \left\|F(\mathbf{m}_j) - F(\mathbf{m}_i)\right\|)), \tag{5.3}$$

depends on how the distance changes between clusters i and j as they are projected from the LUV source space to the target gray space. In order to compare distances between the LUV source space and the target gray space, function $F()$ transforms gray values from the target space to CIE L^*. Then, distances in L^* space are directly comparable to distances in LUV space. For our examples, a gray value $\mathbf{m}_i$ is converted to L^* as follows:

$$F(\mathbf{m}_i) = \begin{cases} \mathbf{m}_i & \mathcal{T} = \text{CIE } L^*, \\ C_{sRGB \to L}(\begin{bmatrix} \mathbf{m}_i & \mathbf{m}_i & \mathbf{m}_i \end{bmatrix}) & \mathcal{T} = \text{luma}, \\ C_{Y \to L}(\mathbf{m}_i) & \mathcal{T} = \text{CIE } Y. \end{cases} \tag{5.4}$$

The function $C_{sRGB \to L}()$ converts an sRGB vector to L^* by converting it to LUV space and returning the L^* component. The function $C_{Y \to L}()$ converts the Y component from CIE XYZ to L^*.

The other components defining a_{ij} in Equation 5.3 are the sigmoidal weight ψ_{ij}, the scale function $S()$, and the parameter k. The sigmoidal weight and scale function are defined as

$$\begin{aligned} \psi_{ij} &= \frac{1}{1 + e^{-\kappa(\left\|F(\mathbf{m}_j) - F(\mathbf{m}_i)\right\| - c)}} \\ S(x_{ij}) &= \frac{\max(0, x_{ij})}{\max_{(i,j) \in \mathcal{E}} x_{ij}} \end{aligned} \tag{5.5}$$

with our sigmoid centered on $c = 15$, determined empirically from our image data. For this application, in order to accommodate various standard

grayscale target spaces with different ranges, the user provides a parameter k_p expressed as a percentage of the target gray range instead of directly supplying k. For our examples, the parameter k is then calculated from k_p as

$$k = \begin{cases} 100 \cdot k_p & \mathcal{T} = \text{CIE } L^*, \\ k_p & \mathcal{T} = \text{luma}, \\ k_p & \mathcal{T} = \text{CIE } Y, \text{ with } Y \in [0,1]. \end{cases} \tag{5.6}$$

Critical pairs are pairs of clusters that have large color differences but similar gray values. In Figure 5.1, for example, the two clusters representing the red sun and its neighboring blue background would form a critical pair because the clusters' gray values are very similar despite their strong color contrast. The critical pairs indicate where contrast enhancement is most needed, and we use this information to assign the per edge and per cluster weights to direct the enhancement to these areas. Critical pairs are determined as outlined in Section 4.4 and are based on whether clusters overlap each other and by how much.

We solve the linear least squares problem in Equation 5.1 for the optimal gray cluster means within the target gray space. The target space bounds $[lb, ub]$ are $[0, 1]$ for luma and CIE Y and $[0, 100]$ for L^*. We make sure the output gray cluster means are within these bounds by constraining each cluster's output $\mathbf{x}_i$ to stay within $[lb + r_{m_i}, ub - r_{m_i}]$, where r_{m_i} is cluster i's radius in the target gray space.

5.2.5 Blending and Final Projection

The optimization solves for optimal gray values for the cluster means, translating the clusters from their initial gray values. These optimization results must be transferred back to the pixels, yielding the output gray image. Each pixel's initial gray value is shifted by a weighted blend of cluster movements, as described in Section 4.5. The soft blending helps avoid quantization-like discontinuities. The output pixel values are clipped to the bounds of the target gray space ($[0, 1]$ for luma and $[0, 100]$ for L^*).

5.2.6 Display

Some grayscale target spaces are not accepted by the hardware for direct display, so, in these cases, we must convert our output gray image to display ready values. For target grayscale spaces that are linear with luminance, we make our results suitable for display on an sRGB monitor by accounting for a display gamma of 2.2. CIE Y, a representation of luminance, is an

example of one of these spaces. For these linear spaces, we would scale the target gray range to $[0, 1]$ and then correct for the display gamma by raising the gray values to the power $\frac{1}{2.2}$. The final step would be to linearly scale the values to $[0, 255]$ and quantize, storing each pixel in 8 bits. Luma and L^*, the target gray spaces we used, are nonlinear with luminance. Luma is already suitable for display on an sRGB monitor, and L^* has a cube root relationship with relative luminance similar enough to the power $\frac{1}{2.2}$, so we do not modify them except to linear scale values to $[0, 255]$ for 8-bit storage.

5.3 Results

Our method is able to preserve the contrasts lost during a standard projection to grayscale while remaining close to the standard gray, which is important for creating a plausible grayscale result. For example, in Figure 5.4 the chromatic contrast our method restores the contrast between the red and blue elements of the impressionist painting is lost during the standard L^* conversion to grayscale. Our method restores this contrast as grayscale contrast, similar to other color to gray methods. The rings of candy in Figure 5.5 lose most of their contrast during the standard luma conversion. Our method enhances the differences between the rings without moving too far from the initial luma values. In the luma image, the yellow ring (ring 3) is slightly lighter than its green and red neighbors (rings 2 and 4), and our result preserves this ordering. This resemblance to the luma image results in a more realistic grayscale version than global mappings such as those from Rasche et al. [Rasc 05b] and Grundland and Dodgson [Grun 07]. In their output images, the green ring is lighter than the yellow ring. This intensity reversal and the large deviation from the standard luma image make these output images less credible grayscale versions of the original image than our result. Our optimization approach enables us to simultaneously preserve contrast while staying close to the initial mapping.

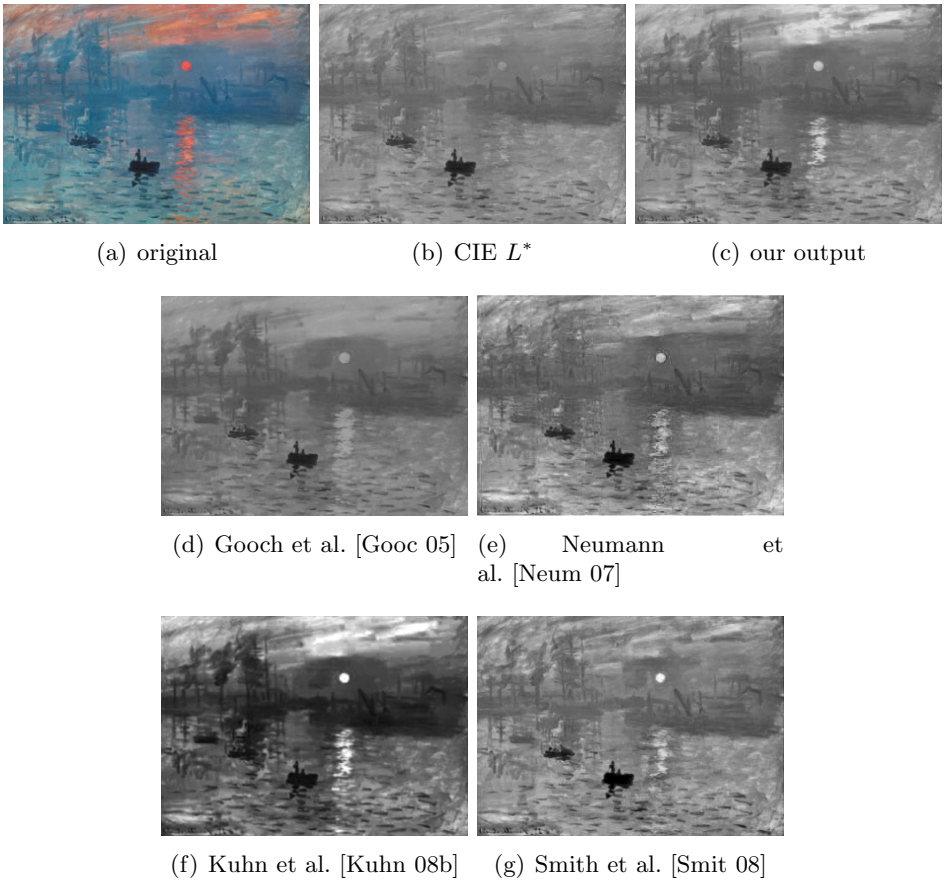
Working at the cluster level helps us overcome limitations with local filtering methods and gradient based methods. Figure 5.6 shows how our method avoids halo artifacts that arise in methods with local filtering operations such as Smith et al. [Smit 08]. The island in this map image disappears when converted to luma. Both our method and the method of Smith et al. restore the island visibility, but Smith et al.'s method introduces a halo around the island due to its local unsharp masking step. Gradient based methods can create color ramps, or gradients, that are not in the original image. By only considering relationships between neighboring pixels, these

methods could introduce color ramps in solid, single color regions. Figure 5.7 shows an example of these artifacts in the gradient based method of Alsam and Drew [Alsa 08] and shows that our method is not as susceptible to them. Our cluster-based method considers relationships between clusters instead of pixels and makes modifications at the cluster level so that all pixels within a cluster receive similar modifications. Through our semi-local, cluster-based approach we can preserve local contrast, stay close to an initial mapping, avoid halo artifacts, and prevent unwanted color ramps in local neighborhoods.

Figure 5.8 contains more results for color to gray conversion.

5.4 Summary

The problem of converting color images to grayscale fits within our framework. It has a few standard grayscale projections (e.g., luma, CIE L^*, and CIE Y) that fill the role of the initial mapping. This initial grayscale projection simultaneously imposes a loss of information as three dimensions are reduced to one and serves as a standard grayscale rendition of the color image. We apply our general method to color to gray conversion, achieving the goals of the framework. Our method enhances local grayscale contrast where color contrasts were lost during the initial grayscale projection without straying too far from that standard projection. As with the other applications in the framework, our results for color to gray conversion preserve local contrasts from the original image while maintaining the natural colors given by the standard projection.

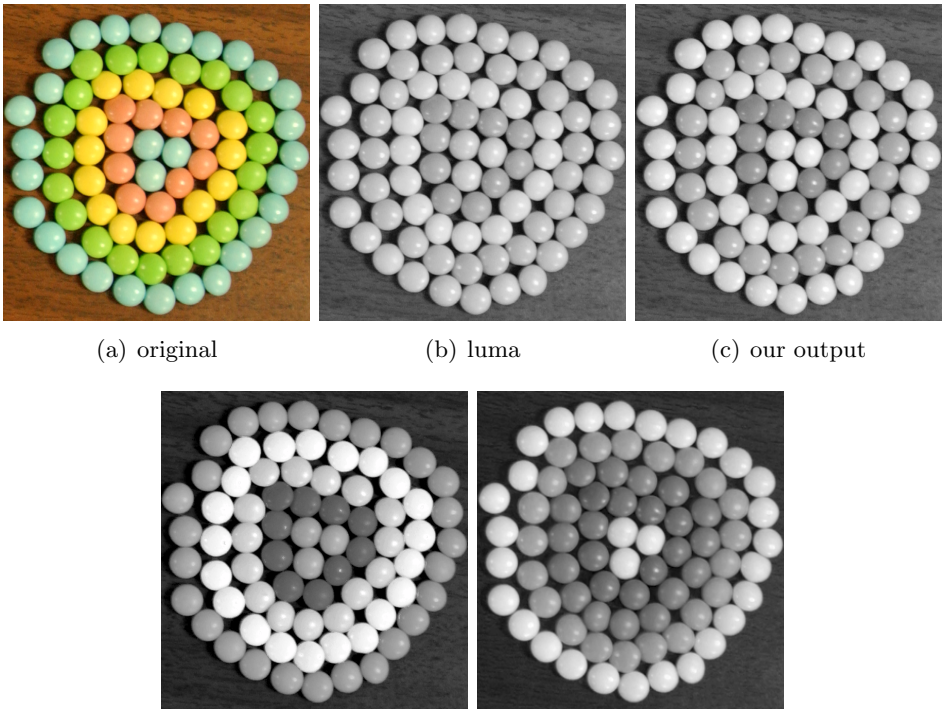


(a) original (b) CIE L^* (c) our output

(d) Gooch et al. [Gooc 05] (e) Neumann et al. [Neum 07]

(f) Kuhn et al. [Kuhn 08b] (g) Smith et al. [Smit 08]

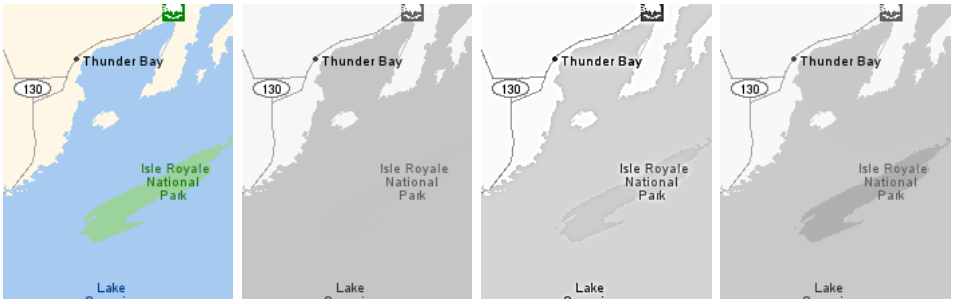
Figure 5.4: **Preserving contrasts.** Our output (c) preserves the contrast between the red and blue elements in the original image (a) lost during the standard L^* projection (b). Our method preserves the contrast between the sun and the background, similar to other color to gray methods (d-g) [Gooc 05, Neum 07, Kuhn 08b, Smit 08]. *Original image courtesy of Gooch et al. [Gooc 05].*



(a) original (b) luma (c) our output

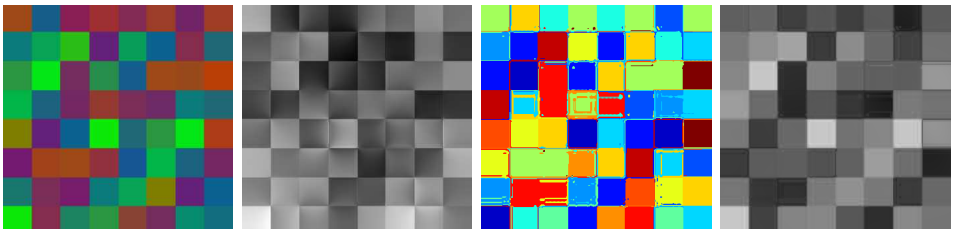
(d) Rasche et al. [Rasc 05b] (e) Grundland and Dodgson [Grun 07]

Figure 5.5: **Maintaining naturalness.** During the standard projection to luma, the rings of candy lose contrast. Our method enhances contrast between the rings without deviating too far from the luma image, producing a more realistic grayscale version than global mappings like those of Rasche et al. [Rasc 05b] and Grundland and Dodgson [Grun 07]. Note: Grundland and Dodgson's output (e) has a lower resolution since their input image was a downsampled version of the original image. The original resolution was downsampled from 596×616 to 300×300. *Original image courtesy of Rasche et al. [Rasc 05b].*



(a) original (b) luma (c) Smith et al. [Smit 08] (d) our output

Figure 5.6: **Avoiding halos.** The green island disappears when the original image (a) is converted to luma (b). Our method and the method of Smith et al. [Smit 08] both restore contrast, but our method does not contain halo artifacts around the island that are common to methods with local filtering. *Original image courtesy of Yahoo Maps!/NAVTEQ/DigitalGlobe.*



(a) original (b) Alsam and Drew [Alsa 08] (c) cluster image (d) our output

Figure 5.7: **Avoiding gradient artifacts.** When converting the original image in (a) to grayscale, the gradient based method of Alsam and Drew [Alsa 08] shown in (b) introduces color ramps, or gradients, within the single colored blocks. Gradient artifacts are visible where pixels near the edges are different intensities from pixels in the centers of solid color blocks. Our cluster-based method groups all pixels within a block into a single cluster and modifies each block as a whole to avoid these artifacts, producing the output in (d). Image (c) shows the clustering results where each color is a different cluster. Note: The images (a) and (b) extracted from Figure 3 in Alsam and Drew [Alsa 08] have some compression artifacts, so our method was not run on the actual original image but rather the compressed image. This is why the clusters are not perfect squares and why these artifacts are present in our output image. Nevertheless, this example shows that our method applies similar modifications to pixels within a cluster. *Original image courtesy of Alsam and Drew [Alsa 08].*

(a) original (b) CIE L^* (c) our output

Figure 5.8: **More results.** These are more results for color to gray conversion. *Original images (a) courtesy of (row 1) Grundland and Dodgson [Grun 07] and (row 2) Leon Bli.*

Chapter 6

Gamut Mapping

6.1 Introduction

Gamut mapping is important for reproducing images on different devices such as displays, cameras, projectors, and printers. It is often desirable to display or print an image on one device and have it look the same as it did on its original device. However, the reproductions might not exactly match their originals in appearance because different devices have different gamuts. A gamut is the set of colors the device can reproduce as determined by the device's technology. Typically, a printer's gamut is smaller than a display's gamut, leading to the printer's inability to reproduce some display colors. Cross media reproduction is a process that aims at preserving color appearance in reproduced images, and gamut mapping is the part of the process that accounts for the differences between source and target gamuts.

Gamut mapping converts colors in an image from a source gamut to a target gamut. The goal is to create a mapping such that the reproduced image viewed in the target gamut best matches the color appearance of the image viewed in the source gamut. Color appearance is affected not only by different gamuts but also by different viewing environments. The cross media reproduction pipeline, depicted in Figure 6.1, involves device characterization steps and viewing environment transforms to put an input image into a common color space such as CIE XYZ. Within this common space, gamut mapping maps colors from the source gamut to the target gamut. A gamut comprises a set of colors which can be visualized as a volume in the common color space. Figure 6.2 shows the different volumes for an example pair of source and target gamuts. In-gamut colors exist where the volumes intersect, and these source colors are reproducible in the target gamut. Out-of-gamut regions, where the target gamut does not overlap the source gamut, contain source colors that will not be reproducible in the target gamut. During gamut mapping, out-of-gamut colors need to be mapped to colors within the target gamut. Once source colors are mapped to colors within the target gamut, cross media reproduction proceeds with inverse transforms to account for viewing environment and device characterization

steps to prepare for display on the output device. In this thesis, we limit our work to the gamut mapping step for gamut reductions.

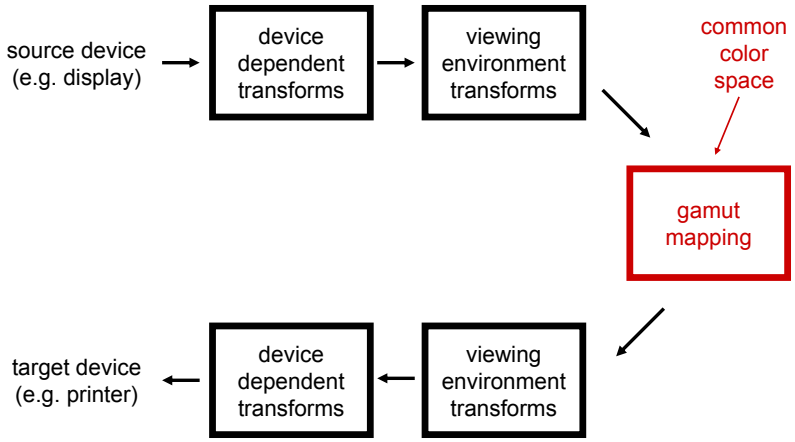


Figure 6.1: **Cross media reproduction pipeline.** Given an input image defined within the space of one device, the goal of the cross media reproduction pipeline is to create a color accurate reproduction of the image on a target device. This includes accounting for device characteristics, viewing conditions, and gamuts. One example usage would be to prepare an image shown on a display for printing. The image is converted from a device-dependent space to a device-independent space, and a color appearance model is applied to account for original viewing conditions. The image is now in a common color space where gamut mapping takes place to map the image from the source gamut to the target gamut. From here, the reverse steps are taken, transforming the image for the target viewing conditions and into the target device's space.

When gamut mapping an image to a smaller target space, details in the out-of-gamut regions are lost. For example, details in the red feathers of the bird in Figure 6.3 are lost during a standard mapping to a smaller toy gamut, and the feathered region appears smooth instead of textured. The standard mapping used here is Hue Preserving Minimum ΔE (HPMINDE)

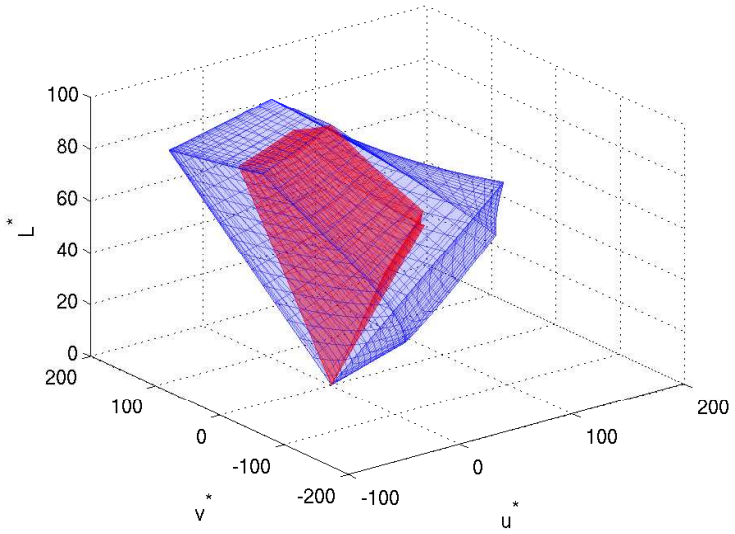
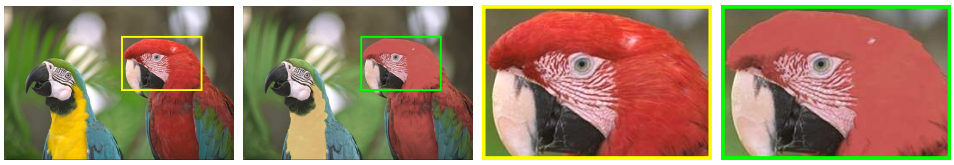


Figure 6.2: **Gamuts.** A gamut is the set of reproducible colors of a device and can be visualized as a volume in 3D space. The toy gamut (red), which we use as a target gamut in many of our examples, is smaller than the sRGB gamut (blue). They are shown here in CIE LUV space.

clipping which keeps in-gamut points as they are and maps out-of-gamut points to their closest points on the gamut boundary within a plane of constant hue. Figure 6.4 shows a 2D slice through the gamut volume along a plane of constant hue in LCH space and illustrates how out-of-gamut points are mapped to the gamut boundary. With this mapping scheme, different out-of-gamut points along the same mapping vector will map to the same color on the target gamut boundary, and the contrast between them will be lost. Another standard mapping, Sigmoidal Gaussian Cusp Knee (SGCK) compression, tries to preserve these contrasts by compressing all colors into the gamut at the expense of color accuracy. With SGCK, color accuracy of the in-gamut colors is sacrificed because both in-gamut as well as out-of-gamut colors are modified and compressed along mapping lines. Gamut mapping methods strive to preserve both contrast and color accuracy, two competing qualities.

Gamut mapping fits within our framework because there is a loss of information when mapping from a source space to a target space. In gamut map-



(a) original (b) HPMINDE mapped (c) original, zoomed (d) HPMINDE mapped, zoomed

Figure 6.3: **Loss of detail.** The original sRGB image in (a) is mapped to a smaller toy gamut with less saturated primaries via HPMINDE, a standard gamut mapping method, in (b). Since the red feathers of the bird are out-of-gamut, the mapping causes a loss of detail in the zoomed in regions in (c) and (d). *Original image courtesy of Kodak [Koda].*

ping, a tristimulus color image is mapped to another tristimulus color image that contains only colors from the target space. Even though $n = m = 3$ within the context of our n to m dimensional mapping framework, the source and target spaces fill different volumes in $\mathbb{R}^3$ and are represented by different constraints. The mismatch in volumes leads to a loss of details in the out-of-gamut regions. Similar to the other problems in the framework, in the extreme case of information loss, gamut mapping exhibits metamerism when different colors map to the same color in the target gamut as diagrammed in Figure 6.4.

By applying our method, outlined in Chapter 4, to gamut mapping we develop a new local gamut mapping method that preserves details in the out-of-gamut regions yet remains close to the HPMINDE mapping for color accuracy. Our first goal is to preserve the contrasts between out-of-gamut colors. This is important for preserving details in out-of-gamut regions such as the texture of the red feathers on the bird in Figure 6.3. To preserve these contrasts, we must allow some colors to deviate from their most accurate mapping, but the colors should not stray too far or else the output will not be a faithful reproduction of the original image. Therefore, our second goal is to preserve color accuracy by keeping output colors close to the HPMINDE mapping. For gamut mapping, it is also important to prevent hue shifts and chromatic additions to achromatic colors since they cause particularly disturbing artifacts. Overall, our gamut mapping method aims to simultaneously preserve contrasts and colors from the source image. These two goals match the goals of our framework, which are to preserve contrasts from the source space while staying close to a natural mapping. To fur-

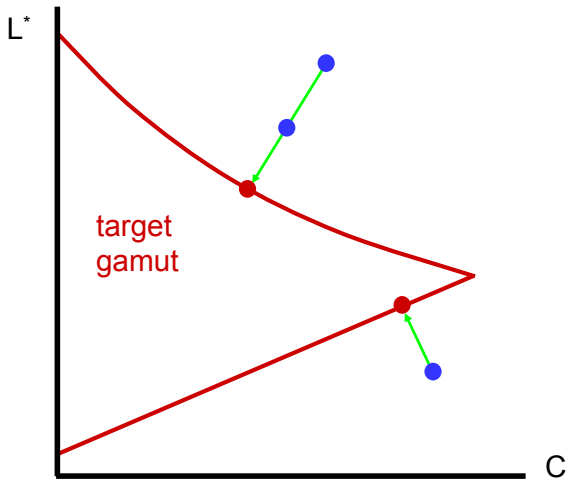


Figure 6.4: **HPMINDE mapping.** This is a 2D slice through the gamut volume along a plane of constant hue in LCH space. HPMINDE maps out-of-gamut points (blue) to their closest points on the target gamut boundary (red). It is possible for different out-of-gamut colors to map to the same color on the boundary, resulting in a loss of contrast between the colors in the mapped image.

ther preserve colors, we add terms to the optimization to preserve hue and achromaticity.

Our gamut mapping results show how our method is able to preserve details in the out-of-gamut regions without deviating too far from the natural HPMINDE mapping. Unlike many spatial gamut mapping methods, our semi-local approach modifies local areas to enhance contrast instead of using local filtering techniques which often lead to halo artifacts. During the standard HPMINDE mapping, it is possible for the lightness contrast between two colors to be inverted. Since our optimization for gamut mapping tries to make vectors between clusters match their original directions in the source space, our method fixes these lightness reversals. As with the other applications in the framework, we are able to produce gamut mapping results that preserve contrasts lost during the standard mapping while staying close to it for a faithful reproduction of the original image within the constraints of the target space.

6.2 Method

We apply the general method described in Chapter 4 to gamut mapping. In this section, we explain the method in terms of gamut mapping and include application-specific details. First, we cluster together pixels that are similar in color and spatial coordinates. Then, using a constrained optimization, we solve for optimal cluster colors within the target space. Finally, we transfer the output cluster colors back to the pixels in the blending step.

For gamut mapping, the pipeline diagram from Figure 4.1 still applies, but the initial mapped image is not needed to compute cluster statistics or to compute output pixel values in the blending step. Figure 6.5 shows the modified pipeline for gamut mapping. There are two differences. First, instead of calculating the mapped cluster means from the mapped image, we calculate them by projecting the original cluster means to the target space. Second, during the blending step, the output pixel values are calculated by modifying the original pixel values instead of the mapped pixel values. This helps preserve intra-cluster detail and can only be done when the source and target spaces have the same number of dimensions, such as in gamut mapping.

6.2.1 Projection to Target Space

Our input images are typically represented in sRGB space or CIE XYZ space. In any case, we losslessly convert our input images to a perceptually uniform space. Specifically, we choose to represent our input images in CIE LUV. The source space $\mathcal{S}$ is a subset of CIE LUV space, but the exact constraints of $\mathcal{S}$ (in $\mathbb{R}^3$) are irrelevant to our method. It is the constraints of the target space $\mathcal{T}$, a smaller subset of CIE LUV space, that are needed by our method.

To achieve a color accurate result, we use HPMINDE clipping as the standard projection from $\mathcal{S}$ to $\mathcal{T}$. The initial mapped image produced by HPMINDE serves as our color accurate reproduction of the original image, but it suffers from a loss of details in the out-of-gamut regions due to clipping of these regions. Our method aims to restore the lost details without straying too far from the natural colors of the HPMINDE mapping.

Target Space Constraint Representation

For gamut mapping, we need to be able to constrain mapped colors to the target gamut. The target gamut $\mathcal{T}$ is often not axis-aligned and must be

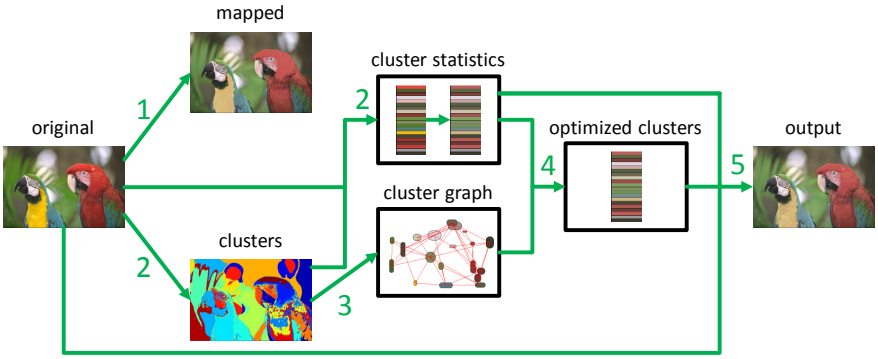


Figure 6.5: **Method overview for gamut mapping.** 1) We project the original image to the target space to get the initial mapped image. 2) We cluster the pixels in spatio-chromatic space and compute cluster statistics. The cluster image as well as original and mapped cluster means are shown here. The original cluster means are projected to the target space, yielding the mapped cluster means. 3) We create a graph by connecting spatially close clusters. 4) We optimize for new cluster colors in the target space. 5) We transfer the optimization results back to the pixels with a blending step. For gamut mapping, the original pixels are modified to obtain the output pixels. *Original image courtesy of Kodak [Koda].*

represented as a set of constraints describing a finite volume in $\mathbb{R}^m$. We represent the target gamut in a voxel grid whose voxels mark the gamut boundary. This approach enables us to compute signed distance fields from the boundary, which are useful for mapping clusters to the boundary and in turn enforcing the constraints in the optimization.

As with the source space, we continue to work in a perceptually uniform space for the target space, from which we can then map into the actual target space with a lossless transform. We represent the target gamut with a voxel grid in LCH space, the polar equivalent of CIE LUV. The initial HPMINDE mapping from $\mathcal{S}$ to $\mathcal{T}$ operates within planes of constant hue, which are easily represented in this polar space. In our method, we generalize the HPMINDE mapping which uses the ΔE^*_{ab} metric from CIE LAB to extend to CIE LUV, a similar perceptually uniform space, by using the ΔE^*_{uv} metric instead. This means that we use L_2 distances within constant hue planes in LCH space to map colors. For a given color, we retrieve its constant hue plane, shown in Figure 6.6, and map the color to the closest point on

the gamut boundary using L_2 distance. The standard HPMINDE process applies this mapping independently to each pixel in the source image. As a result, many similar colors outside the gamut will be mapped directly onto the boundary, which results in a loss of texture detail. Instead, we propose a new cluster-based mapping that preserves structure within such a group of colors. A cluster with radius r is mapped inside the gamut boundary by mapping the cluster's mean to the closest point on the distance field's level set that is r away from the boundary. To prevent the cluster from crossing the lightness axis, points that are $< r$ away from the L^* axis are excluded from the $-r$ level set, and source clusters that already intersect the L^* axis are constrained to map to colors with the same C^* value.

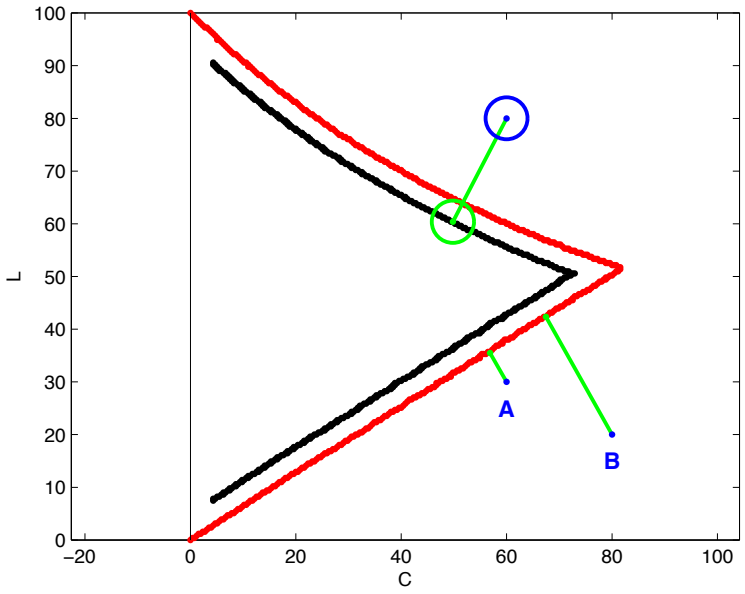


Figure 6.6: **Our HPMINDE mapping and cluster mapping.** 2D slice through gamut along a plane of constant hue. Original points (blue) are mapped to their closest points (green) on the gamut boundary (red). To map a cluster with radius r, we precompute a signed distance field for the gamut boundary and map the cluster to the level set $-r$ (black). It is possible for the HPMINDE mapping to invert lightness values L for a pair of points (A and B).

Voxel Grid. We create our voxel grid to have 256 voxels in the L^* dimension, resulting in a voxel size of 100/255 in the L^* and C^* dimensions. We divide the LCH space into 360 hue planes. We extend the number of voxels in the C^* dimension to circumscribe the rectangular LUV bounds that include the max chroma point of our largest source gamut. To fill the voxel grid, we mark voxels that contain the gamut boundary. For example, to create a voxel grid for sRGB, voxels containing colors on the sRGB cube surface are marked as boundary voxels. To make a thin, closed boundary for each hue plane, only the max chroma boundary voxel in each L^* row is kept, and the rest of the boundary is filled in by interpolation, making sure that each L^* row and each C^* column contains a boundary voxel.

6.2.2 Clustering

To separate our image into local areas of similar colors, we cluster pixels as described in Section 4.2. We augment our LUV source image with weighted spatial dimensions and cluster the augmented pixels with feature vectors (L, U, V, x', y') in spatio-chromatic space $\mathcal{U}$ in $\mathbb{R}^5$. The contrast between the clusters represents the local color contrast between spatially neighboring areas, and it is this contrast that our optimization seeks to preserve. We calculate cluster means and covariance matrices for the clusters in $\mathcal{U}$, which also gives us these statistics for the source clusters in $\mathcal{S}$ in LUV. Radii for the source clusters are computed and used along with the covariance matrices to define spherical and ellipsoidal shape for both source clusters in $\mathcal{S}$ and mapped clusters in $\mathcal{T}$. Since the target space and the source space are both in $\mathbb{R}^3$, we use the same radii for mapped clusters as we do for the source clusters, so $r_m = r_o$ which we refer to without the "m" and "o" subscripts in this chapter. The cluster means for the mapped clusters in $\mathcal{T}$ are obtained by projecting the source clusters to the target space, using our cluster mapping described in Section 6.2.1, such that the approximate cluster spheres fit within the target gamut.

6.2.3 Graph Creation

We create the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of clusters, and the edges in $\mathcal{E}$ connect spatially close clusters. The edges are determined by the dilation-and-neighbor-counting procedure described in Section 4.3.

6.2.4 Optimization

During the optimization step, we solve for new LUV cluster colors in the target space $\mathcal{T}$ that preserve local contrast between clusters. To create a faithful reproduction of the original image, the new cluster colors should also be close to the initial HPMINDE cluster mapping, should not shift in hue, and should not be chromatic if the original color is achromatic. Since the new cluster colors need to be within the constraints of the target space, we use the constrained optimization procedure described in Section 4.4.2. We solve for the optimal cluster colors within LUV space in $\mathbb{R}^3$, project these colors to the target gamut $\mathcal{T}$, and iterate. We run $Z = 20$ iterations in our implementation. To solve for the optimal cluster colors $\mathbf{x}^{(z)}$ in LUV space, we minimize Equation 4.11, repeated here along with Equations 4.2, 4.5, 4.12, 4.7, and 4.8, modified to include the iteration number z.

$$\mathbf{x}^{(z)} = \arg\min_{\mathbf{x}^{(z)}} E_T + \lambda_M E_M + \lambda_H E_H + \lambda_L E_L + \lambda_P E_P, \qquad z = 0, \ldots, Z-1 \tag{6.1}$$

$$E_T = \sum_{(i,j)\in\mathcal{E}} \tau_{ij}((\mathbf{x}_j^{(z)} - \mathbf{x}_i^{(z)}) - \mathbf{t}_{ij})^2$$

$$E_M = \sum_{i\in\mathcal{V}} \tau_i(\mathbf{x}_i^{(z)} - \mathbf{m}_i)^2$$

$$E_H = \left\| H\mathbf{x}^{(z)} \right\|^2$$

$$E_L = \left\| L(\mathbf{x}^{(z)} - \mathbf{m}) \right\|^2$$

$$E_P = \begin{cases} 0, & z = 0, \\ \sum_{i\in\mathcal{V}} (\mathbf{x}_i^{(z)} - \mathbf{p}_i^{(z-1)})^2, & z = 1, \ldots, Z-1 \end{cases}$$

The target term E_T preserves original color contrast between clusters. The regularization term E_M, hue term E_H, and achromaticity term E_L keep the optimized colors close to the initial HPMINDE mapping, prevent hue shift, and maintain achromaticity, respectively. The term E_P helps the optimization converge to a solution within the constraints of the target gamut. These terms and their associated parameter weights are discussed in Section 4.4.

Since $n = m$ and the original and mapped clusters are both in $\mathbb{R}^3$, we can set our targets to be the exact vectors we wish to preserve. Instead of using Equation 4.3 to define our target vectors as mapped vectors extended by some additional magnitude, we set the target vector $\mathbf{t}_{ij}$ for edge (i, j) to be the original difference vector $\mathbf{o}_{ij}$ between the clusters in the source space. This lets us preserve original contrasts in both direction and magnitude.

The target vector is defined as

$$\mathbf{t}_{ij} = \mathbf{o}_{ij}. \tag{6.2}$$

Critical pairs, or pairs of clusters that have lost significant contrast due to the initial HPMINDE cluster mapping, are found in order to calculate weights τ_{ij} and τ_i. We find critical pairs by testing if the initial mapping causes clusters to overlap more in the target space than in the source space. In Section 4.4, the general procedure compares fractional overlap amounts, but for the case of gamut mapping, this reduces to comparing nonfractional overlap amounts $\upsilon_{o_{ij}}$ and $\upsilon_{m_{ij}}$. This is because we use the same radii for the source and target spaces, i.e. $r_m = r_o$. Edge (i, j) is critical if Equation 4.10 is true, and this condition reduces to

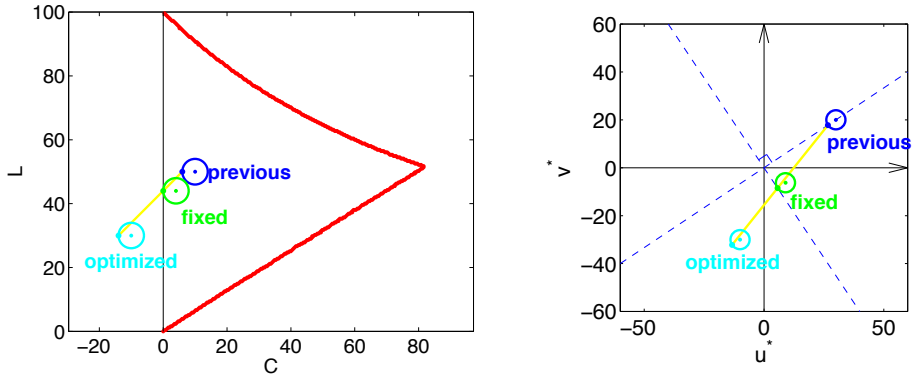
$$N(\upsilon_{m_{ij}}) > N(\upsilon_{o_{ij}}), \tag{6.3}$$

where $N(x) = |\min(x, 0)|$.

When solving for the optimal cluster colors $\mathbf{x}^{(z)}$ by minimizing Equation 6.1, we constrain the clusters to be within LUV bounds, where L^* is in $[0, 100]$, u^* is in $[-141, 243.5]$, and v^* is in $[-139, 125.5]$. We ensure that each cluster's mean is at least r_i away from the LUV bounds, where r_i is the radius of cluster i.

In gamut mapping, it is important that we do not let a cluster cross the lightness axis during the optimization to avoid hue changes. The hue term is a soft constraint that prevents the cluster from moving too far off the geometric plane through the L^* axis and the cluster's original mean. However, the geometric plane contains two "hue planes" separated by the L^* axis with the two hue angles 180° apart from each other. The hue term will not prevent a cluster from crossing the L^* axis and moving into its complementary hue plane. To avoid this, after the least squares minimization, we project any clusters that have crossed the lightness axis back across the axis to the side of its original hue plane. Figure 6.7 shows how we move a cluster back across the lightness axis. In Figure 6.7(a), the cluster has crossed the L^* axis into its complementary hue plane. We find a point on the cluster's surface r from the mean, and we move this point to the intersection of the L^* axis and the line between the cluster's current location and its location during the previous iteration. Since the hue term is only a soft constraint, optimized cluster locations might not be exactly on the geometric hue plane. In Figure 6.7(b), the u^*v^* plot shows a top down view of a similar situation as Figure 6.7(a) but with a cluster that is not exactly on its original hue plane. In this case, the cluster has moved across the plane perpendicular

to its hue plane, which is similar to crossing the lightness axis. We move the cluster back to its hue's halfspace to the intersection of the plane perpendicular to the hue plane and the line between the previous and current cluster locations. This specific lightness axis constraint only applies to our gamut mapping application as it is important for colors to not switch to their complementary hues.



(a) LC plane view

(b) u^*v^* plane view

Figure 6.7: **Fix lightness axis crossings.** During an optimization iteration, a cluster could have crossed the lightness axis. We fix this by moving the optimized cluster (cyan) towards the previous iteration's cluster location (blue) along the yellow line until it crosses the lightness axis in (a) or the plane perpendicular to the hue plane in (b). A point on the cluster surface, which is radius r away from the mean in the negative direction of the cluster's original chroma vector, is used to find the location of the fixed cluster (green). Note that the examples in (a) and (b) are two different examples that are exaggerated for illustration purposes. In practice, these adjustments happen on a smaller scale when all clusters involved are much closer to the lightness axis.

After correcting any lightness axis crossings, we project the optimized colors $\mathbf{x}^{(z)}$ to the target gamut with projection operator $P()$. $P()$ maps clusters to the target gamut boundary as described in Section 6.2.1 and gives us the projected output $\mathbf{p}^{(z)}$. After Z iterations, the projected output $\mathbf{p}^{(Z-1)}$ contains the final, optimized locations for the clusters and becomes input to the blending step.

6.2.5 Blending and Final Projection

The blending step transfers the results of the constrained optimization back to the pixels in a manner that does not create quantization-like artifacts. The result of this step is our output image where each pixel's color is inside the target gamut $\mathcal{T}$. As described in Section 4.5, difference vectors are calculated for each cluster describing each cluster's movement, and blended difference vectors are calculated for each pixel determining the output color of each pixel when added to the pixel. However, gamut mapping is a special case where $\mathbb{R}^m = \mathbb{R}^n$ and both the source and target gamuts can be expressed within a common space, CIE LUV in our implementation. Consequently, instead of starting with the initial HPMINDE mapped image in $\mathcal{T}$ and modifying it, we can start with the original image in $\mathcal{S}$ and map it to $\mathcal{T}$ with our difference vectors. The difference vectors $\mathbf{d}_i = \mathbf{p}_i - \mathbf{o}_i$ for each cluster i describe the cluster's movement from its original position $\mathbf{o}_i$ in $\mathcal{S}$ to its projected output $\mathbf{p}_i$ in $\mathcal{T}$ from the constrained optimization step. We compute the blended difference vectors $\mathbf{d}_q$ for each pixel q as described in Section 4.5 and Equation 4.13. Then, we update each pixel by adding its difference vector to its original source pixel $\mathbf{s} \in \mathcal{S}$. We make sure each output pixel is within the target gamut by clipping the updated pixels to $\mathcal{T}$ using HPMINDE. Blending the difference vectors allows us to transfer the optimization results back to the pixels, avoiding quantization-like discontinuities. By modifying the source image, we are able to preserve intra-cluster detail, or more specifically, detail present in the source cluster but lost during the standard mapping to $\mathcal{T}$.

6.2.6 Display

Our method produces an output image in LUV space within the target gamut $\mathcal{T}$ which we convert to sRGB for display. The target gamuts we use are either equivalent to or contained within the sRGB gamut. This conversion to sRGB space first involves a transformation to linear XYZ space and then to linear RGB space followed by a nonlinear gamma correction. This nonlinear transformation accounts for a display gamma of 2.2.

6.3 Results

Our results show that our gamut mapping method is able to preserve details from the out-of-gamut regions while staying close to the initial HPMINDE mapping for a color accurate reproduction. To illustrate our method on a

conventional display, we map our sRGB source images to a toy gamut with less-saturated primaries than sRGB. The chromaticity diagram in Figure 6.8 contains the primaries of this target gamut inside the triangle formed by the sRGB primaries. Our results in Figures 6.9 and 6.10 contain details in the out-of-gamut regions that were clipped by the standard HPMINDE mapping. The feather texture on the red bird, wrinkles in the hats, and folds in the pink jacket are more visible in our result than in the HPMINDE mapping. Figures 6.14 and 6.15 contain more of our gamut mapping results. In Figure 6.11 we compare our result to that of the SGCK mapping, a globally compressive mapping function. The side of the red barn is in-gamut, but the SGCK mapping compresses these colors to a slightly darker red than the original image. Our result is closer to the color accurate HPMINDE mapping, which has the original, in-gamut colors. As a compressive method, SGCK compresses both in-gamut and out-of-gamut colors, and as a global method SGCK performs the compression whether it helps preserve details in the image or not. While our spatial method also changes in-gamut colors, it has the flexibility to preserve in-gamut colors by looking at spatial neighborhoods and modifying them only when it would help preserve local detail. When mapped to this reduced gamut, our results preserve details from the out-of-gamut regions without trading off as much color accuracy as global compression methods.

As illustrated in Figure 6.6, it is possible for the HPMINDE mapping to invert the lightness of two points. Such a reversal happens for the image of the door in Figure 6.12. In the original image, the areas with chipped paint are lighter than those with paint, but the HPMINDE mapping causes these areas to be darker. Our method is able to fix this lightness inversion since our optimization matches targets that are in the original vectors' directions.

In Figure 6.13 we apply our method to mapping a source image in the larger HP Dreamcolor gamut to the smaller sRGB target gamut, a likely application since many conventional displays today are limited to displaying sRGB content. Our result has more detail in the blades of grass than the standard HPMINDE mapping to sRGB.

6.4 Summary

Gamut mapping fits within our framework as an application to which we apply our general method. Though there is no dimensionality reduction, the information loss in the gamut mapping problem comes from mapping from one gamut to another, each with different constraints within the same

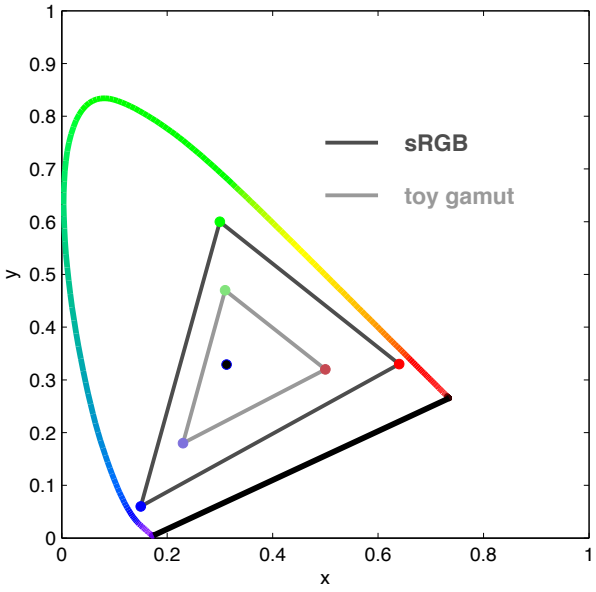


Figure 6.8: **Target gamut.** For visualization on an sRGB monitor, many of our results are mapped to a toy gamut with primaries inside the sRGB gamut. The xy chromaticities of the toy gamut primaries are R:[0.50, 0.32], G:[0.31, 0.47], and B:[0.23, 0.18]. Our target gamut has a D65 white point and linearly transforms to CIE XYZ.

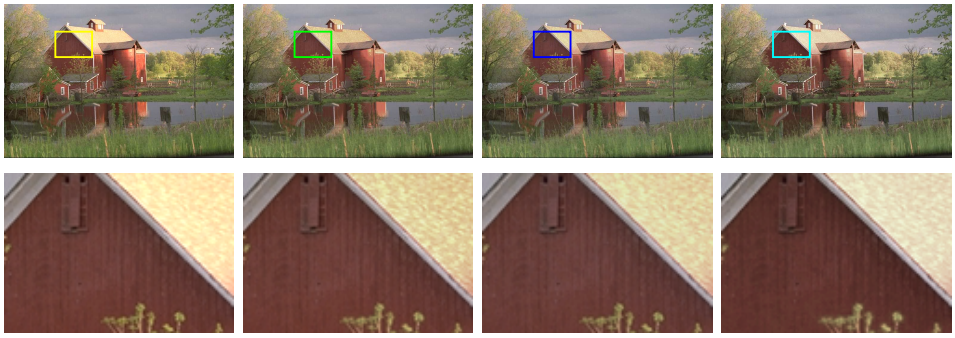
space. The goals of gamut mapping are to simultaneously preserve contrasts and color accuracy. These goals align with the goals of the framework, and we apply our method to achieve them. Our results demonstrate that our method is able to preserve details in the out-of-gamut regions that were lost during the standard HPMINDE mapping while staying close to it for a color accurate reproduction. The spatial aspect of our local method gives us the flexibility to locally tradeoff contrast preservation and color accuracy.

(a) original (b) HPMINDE (c) our output

Figure 6.9: **Preserving details.** When the original RGB images (a) are initially mapped to the smaller toy gamut using HPMINDE clipping, colors are clipped to gamut boundary, causing a loss of detail in (b). Our output (c) is better than the initial mapping at preserving details. The cropped regions, enlarged in Figure 6.10, show the details preserved in the feathers of the bird and in the wrinkles of the hat and jacket. *Original images (a) courtesy of (rows 1 and 2) Kodak [Koda] and (row 3) Fujifilm Electronic Imaging Ltd. (UK) [CIE].*

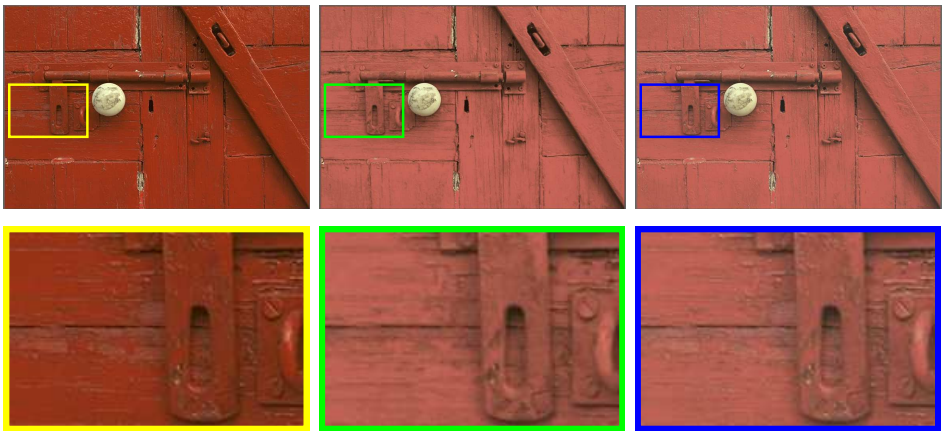
(a) original (b) HPMINDE (c) our output

Figure 6.10: **Preserving details, cropped regions.** This is a close up view of the cropped regions from the images in Figure 6.9. *Original images (a) courtesy of (rows 1 and 2) Kodak [Koda] and (row 3) Fujifilm Electronic Imaging Ltd. (UK) [CIE].*



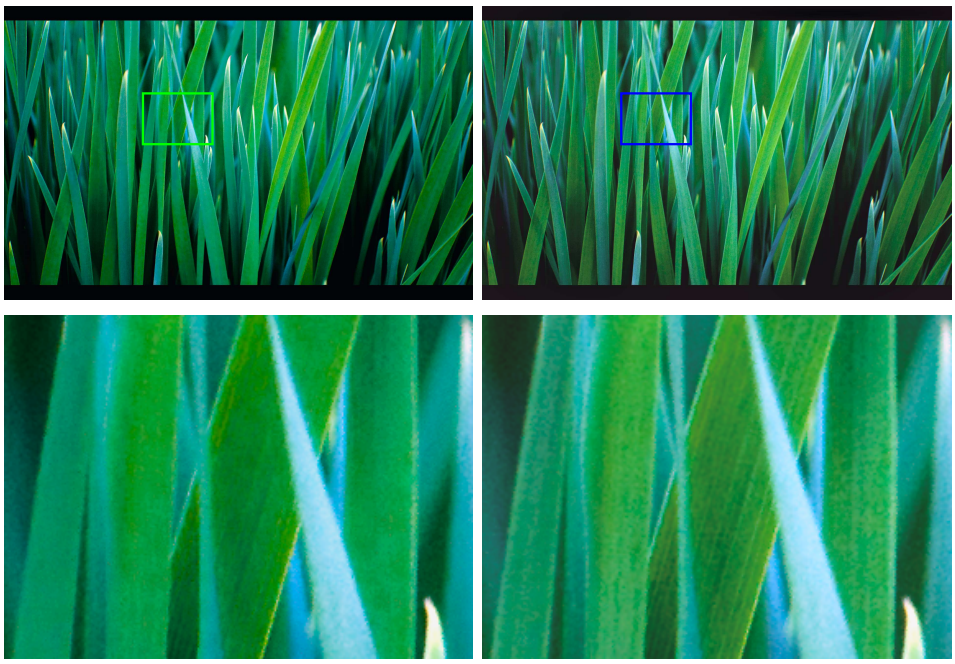
(a) original (b) HPMINDE mapped (c) our output (d) SGCK mapped

Figure 6.11: **Comparison to SGCK.** The side of the red barn in the original image (a) is in-gamut. In (b), the HPMINDE mapping does not modify the in-gamut colors. Our result (c) stays close to this accurate reproduction. In (d), the SGCK mapping globally compresses in-gamut and out-of-gamut colors, resulting in possibly unnecessary color shifts. The change is subtle, but the red barn is slightly darker in the SGCK mapping than in the original image, the HPMINDE mapped image, and our result. No additional detail is gained by modifying this in-gamut region. *Original image courtesy of Kodak [Koda].*



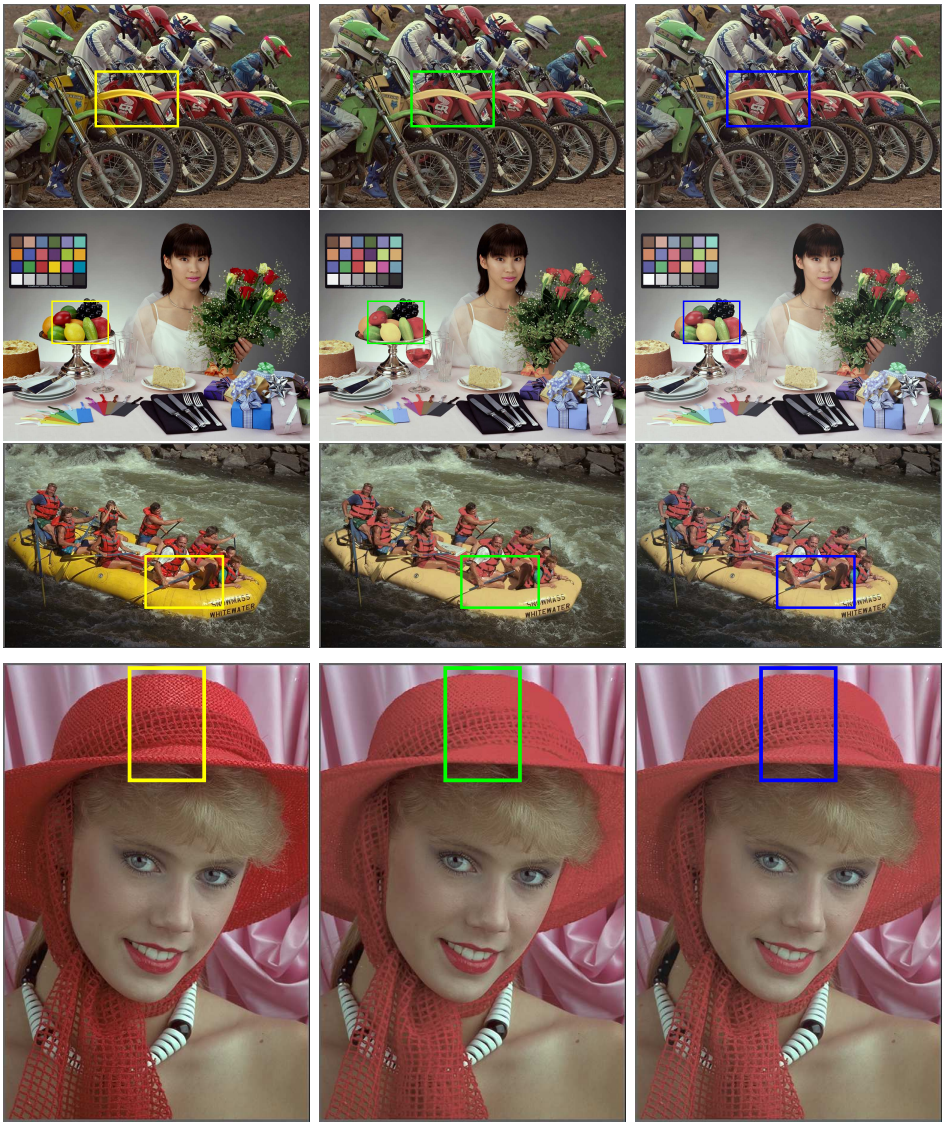
(a) original (b) HPMINDE (c) our output

Figure 6.12: **Fixing lightness inversions.** In the original image (a), the chipped paint areas on the door are lighter than the painted areas. In (b), the HPMINDE mapping maps the two areas to the target gamut in such a way that their lightness relationship is inverted. Our method fixes this inversion since our optimization matches targets that are in the original vectors' directions, producing the output in (c). *Original image courtesy of Kodak [Koda].*



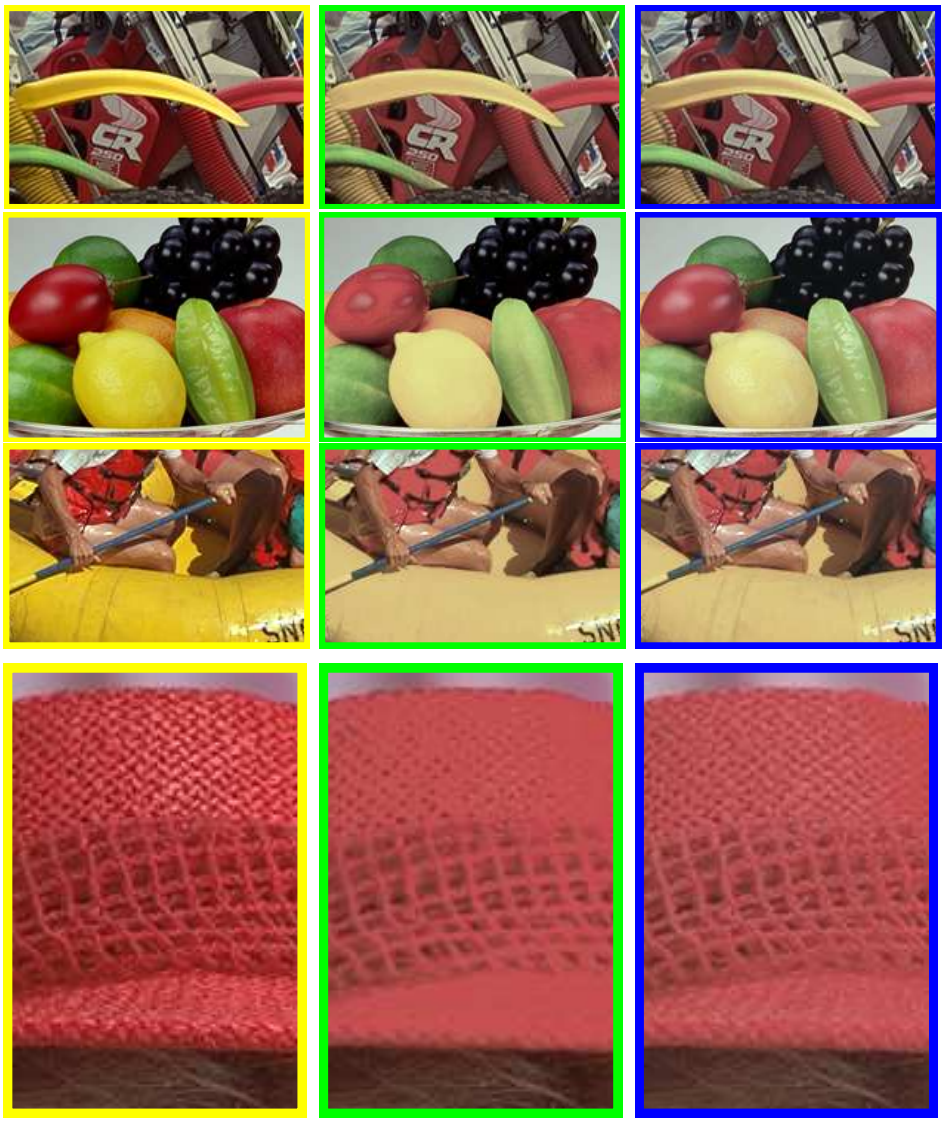
(a) HPMINDE mapped (b) our output

Figure 6.13: **sRGB target gamut.** The original image (not shown) is in the HP Dreamcolor gamut, which is larger than sRGB. We map the image to the sRGB gamut using HPMINDE and our method. Our result (b) contains more details in the blades of grass than the standard HPMINDE mapped image (a). *Original image courtesy of Paul Trepanier.*



(a) original (b) HPMINDE (c) our output

Figure 6.14: **More results.** These are more results for gamut mapping images from sRGB to the toy gamut depicted in Figure 6.8. The cropped regions are enlarged in Figure 6.15. *Original images (a) courtesy of (rows 1, 3, 4) Kodak [Koda] and (row 2) Sony [CIE].*



(a) original (b) HPMINDE (c) our output

Figure 6.15: **More results, cropped regions.** This is a closer view of the cropped regions from the images in Figure 6.14. *Original images (a) courtesy of (rows 1, 3, 4) Kodak [Koda] and (row 2) Sony [CIE].*

Chapter 7

Image Optimization for Color Deficient Viewers

7.1 Introduction

The different types of color deficiencies are characterized by which cones or rods are affected. Rods and cones contain light-sensitive pigments called rhodopsins for rods and opsins for cones. These pigments have specific peak sensitivities and are present in the visual system of a person with normal color vision. Anomalous trichromacy is a type of color deficiency where the absorption spectrum of a cone is displaced from and possibly weaker than the normal opsin sensitivity spectrum. Dichromacy is another type where individuals are missing the pigment for one of the three cones. Dichromats are classified by which cone is affected. The L cones are affected in protanopes, the M cones are affected in deuteranopes, and the S cones are affected in tritanopes. Achromotopsia, a rare color deficiency, happens when individuals having only rods or only one type of cone can only see in shades of one color. In our work, we focus on the dichromatic case where one of the three cones is missing pigments.

Dichromats, with two normally functioning cones out of three, have trouble distinguishing between some colors that people with normal color vision can distinguish easily. For protanopes and deuteranopes, reds and greens can look very similar, and for tritanopes, yellows and blues can look very similar. In Figure 7.1, we simulate how a protanope would see the image of the orange flower, and in the simulated image the large color contrast between the orange flower and the green leaves has been reduced. Methods that optimize images for color deficient viewers aim to help these viewers distinguish between colors that would otherwise look similar.

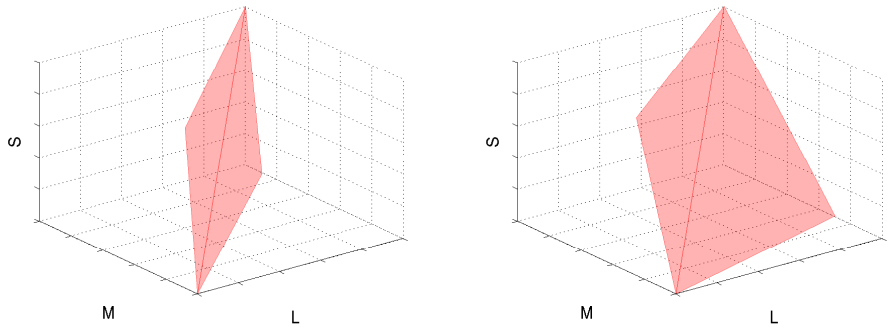
Contrasts are lost or reduced in dichromatic vision because the three dimensions of normal trichromatic vision are reduced to two. The space of distinguishable colors for a particular dichromat can be visualized as a surface in 3D LMS cone space [Bret 97], as shown in Figure 7.2. The surface

(a) original (b) protanope

Figure 7.1: **Dichromatic vision.** The image (b) is a simulation of how a protanope would see the original image (a). Reds and greens appear similar to a protanope, resulting in a reduced contrast between the orange flower and the green leaves. *Original image courtesy of Rasche et al. [Rasc 05b].*

for each dichromat is close to a 2D plane but actually is two half-planes hinged along the neutral axis. To simulate dichromatic vision, Brettel et al. [Bret 97] map LMS colors to the surface along lines parallel to the missing cone's axis. For protanopes, who are missing L cone sensitivities, colors are mapped along lines parallel to the L cone's axis. Similar mappings are done for deuteranopes and tritanopes along lines parallel to the M and S axes, respectively. Without the discriminating coordinate of the third cone, all the colors along a line parallel to the missing cone's axis will map to the same color on the viewer's surface, as shown in Figure 7.3. Colors that are far apart in LMS space can map to colors that are quite close on the color deficient viewer's surface, resulting in a contrast reduction.

Image optimization for color deficient, dichromatic viewers falls within our framework. Naturally as opposed to mathematically, the color deficient viewer experiences a loss of contrast compared to how a viewer with normal vision would see the image. The information loss is physiological; it comes from the viewer's lack of sensitivities in one cone. This problem is actually a specific case of the more general multispectral or multiprimary problem described in Chapter 8. The three LMS cone sensitivity spectra are akin to the spectra of the n bands in a multispectral or multiprimary image. While the cone sensitivities are reduced to two in dichromats, the n bands of a multispectral or multiprimary image are reduced to the three bands in an RGB image. Within the context of the framework, the original $n = 3$



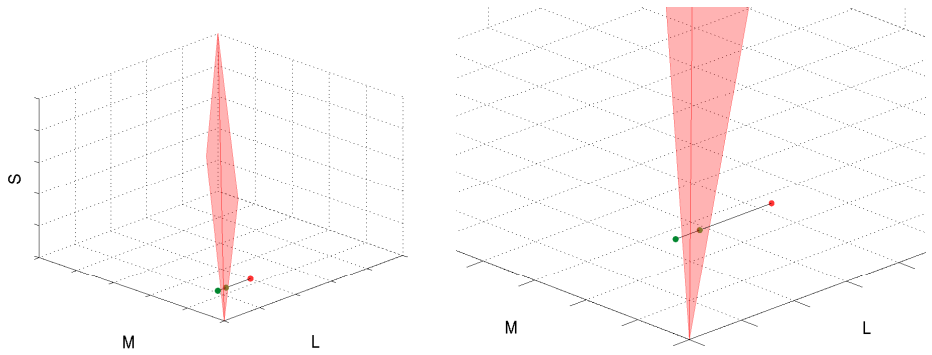
(a) protanopes and deuteranopes (b) tritanopes

Figure 7.2: **Surfaces of distinguishable colors for dichromatic viewers.** The space of distinguishable colors for a dichromatic viewer can be represented in LMS space by a nearly planar surface consisting of two half-planes hinged along the neutral axis [Bret 97]. Brettel et al. [Bret 97] use the same surface for protanopes and deuteranopes (a) and a different surface for tritanopes (b). The L and M cones affected in protanopes and deuteranopes are closer in spectral sensitivity than the S cone affected in tritanopes. Both protanopes and deuteranopes have trouble distinguishing between red and green hues while tritanopes have trouble distinguishing between blues and yellows.

dimensions are mapped to $m = 2$ dimensions. With the loss of dimensionality, as with the other problems in the framework, metamerism occurs when different colors map to the same color on the target surface.

We apply the general method of our framework, outlined in Chapter 4, to recolor images for specific color deficient viewers. Our goal is to reproduce local contrast from the original image, enabling color deficient viewers to distinguish colors that have significant contrast for viewers with normal vision. Our second goal is to retain naturalness by staying close to how the viewer would see the original image. These two goals align with the goals of the framework to simultaneously preserve local contrasts from the source space while staying close to the standard mapping.

Our results show how we are able to recolor images for color deficient viewers, preserving local contrasts and colors within the space of distinguishable colors for the viewer. RGB images are optimized for viewing by either protanopes, deuteranopes, or tritanopes. Our method recolors images with



(a) protanopes (b) zoomed in region of metamers

Figure 7.3: **Simulating dichromacy and metameric colors.** Simulating the vision of a particular dichromat involves accounting for the missing cone type. This is a protanope's surface of distinguishable colors. Brettel et al. [Bret 97] map LMS colors to the 2D surface along lines parallel to the L axis for protanopes. All colors along a mapping line with the same (M,S) values map to the same color on the protanopic surface. These are metamers. The large contrast between the red and green points, which have very different L values, is not seen by a protanope since they have the same (M,S) values and map to the same color. Similar mappings are done for deuteranopes and tritanopes along lines parallel to the M and S axes, respectively, creating metamers out of all points along each line.

colors from each dichromat's surface in a way that enhances local contrast but stays close to what looks natural to them.

7.2 Method

We apply the general method described in Chapter 4 to image optimization for color deficient viewers. In this section, we explain the method in terms of the recoloring problem for color deficient viewers and include application-specific details. First, we cluster pixels together that are similar in color and in the same spatial region. Then, we solve for new colors for the clusters that are optimal for a particular color deficient viewer. Finally, we transfer those new colors back to the pixels in a blending step.

7.2.1 Projection to Target Space

First, we losslessly convert our input color image, which is usually represented in sRGB space, to a perceptually uniform space, allowing us to compare Euclidean distances as measures of perceptual difference. We choose to use CIE LUV as the perceptually uniform source space $\mathcal{S}$.

Since we ultimately want to solve for optimal colors for the color deficient viewer, our target space is the two-cone space of the specific viewer. The target space $\mathcal{T}$ is the MS space for protanopes, the LS space for deuteranopes, and the LM space for tritanopes, each consisting of the two axes corresponding to the two unaffected cone types in each viewer. The natural projection from tristimulus color image to two-cone space would simulate how the color deficient viewer would see the image. We project from $\mathcal{S}$ in $\mathbb{R}^n$ to $\mathcal{T}$ in $\mathbb{R}^m$ by converting LUV source colors to LMS cone space and removing the L, M, or S coordinate corresponding to the affected cone, leaving us in two-cone space. We simulate the color deficiency using Brettel et al. [Bret 97], as shown and described in Figure 7.3. This step takes the 2D coordinates in two-cone space and solves for the missing cone value to project the color onto the surface of the given viewer. Brettel et al. define the neutral axis as the equal energy stimulus of their monitor; we assume our equal energy stimulus is $\begin{bmatrix}1 & 1 & 1\end{bmatrix}^{\mathrm{T}}$ in LMS space. We end up with LMS colors on the viewer's surface, which are the colors they would see. Given the initial mapping of what the color deficient viewer sees when viewing the input image, our optimization aims to recolor the image for that viewer, enhancing contrast where it has been lost due to the missing cone. The goal is to enhance contrast without deviating too far from the natural colors that the viewer initially sees.

7.2.2 Clustering

To separate the image into areas with potential local contrast, we cluster the source image pixels in spatio-chromatic space $\mathcal{U}$ in $\mathbb{R}^5$, as described in Section 4.2. Each pixel's color is augmented with weighted spatial coordinates, forming the feature vector (L, U, V, x', y'), where the spatial weight determines the tradeoff between color and space. Each cluster represents a local area with a single representative color, and the differences between clusters' representative colors are measures of the local contrast between neighboring areas. To get the representative colors and ellipsoid shapes, we calculate cluster means, covariance matrices, and approximate radii for the clusters in $\mathcal{U}$, the source clusters in $\mathcal{S}$, and the mapped clusters in two-cone

space $\mathcal{T}$. The statistics for the mapped clusters come from the pixel data of the mapped image in two-cone space.

7.2.3 Graph Creation

We create the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of clusters, and the edges in $\mathcal{E}$ connect spatially close clusters. The edges are determined by the dilation-and-neighbor-counting procedure described in Section 4.3.

7.2.4 Optimization

The optimization step solves for new cluster colors within the space of the color deficient viewer such that the new colors preserve original color contrast and are close to the colors the viewer naturally sees. For each cluster, we solve for 2D colors in the two-cone space for a particular viewer, which is the target space $\mathcal{T}$. The problem is set up as a linear least squares minimization, solving for the optimal cluster colors $\mathbf{x}$ that minimize

$$\begin{aligned} \mathbf{x} &= \arg\min_{\mathbf{x}} E_T + \lambda_M E_M + \lambda_L E_L \\ E_T &= \sum_{(i,j)\in\mathcal{E}} \tau_{ij}((\mathbf{x}_j - \mathbf{x}_i) - \mathbf{t}_{ij})^2 \\ E_M &= \sum_{i\in\mathcal{V}} \tau_i(\mathbf{x}_i - \mathbf{m}_i)^2 \\ E_L &= \left\| L(\mathbf{x} - \mathbf{m}) \right\|^2 \end{aligned} \tag{7.1}$$

where E_T is the term that preserves contrast, E_M is a regularization term that stays close to the initial mapping to the color deficient surface, E_L is the term that preserves the neutrality of achromatic colors, and λ_M and λ_L are parameter weights. Equation 7.1 is a simplified version of Equation 4.1, and the terms come from Equations 4.2, 4.5, and 4.8. For this application, we have excluded the hue term and the previous iteration term by setting them to zero.

The target term E_T matches the vectors between clusters to some target vectors $\mathbf{t}_{ij}$ for all (i, j) edges in $\mathcal{E}$. The target vector $\mathbf{t}_{ij}$ is described in

Section 4.4 and defined in Equation 4.3, which is repeated here.

$$\begin{aligned} \mathbf{t}_{ij} &= \frac{m_{ij} + a_{ij}}{m_{ij}} \mathbf{m}_{ij} \\ a_{ij} &= k \cdot S(\psi_{ij}(o_{ij} - \left\| F(\mathbf{m}_j) - F(\mathbf{m}_i) \right\|)) \\ \psi_{ij} &= \frac{1}{1 + e^{-\kappa(\left\| F(\mathbf{m}_j) - F(\mathbf{m}_i) \right\| - c)}} \\ S(x_{ij}) &= \frac{\max(0, x_{ij})}{\max_{(i,j) \in \mathcal{E}} x_{ij}} \end{aligned} \tag{7.2}$$

During the projection to the target space, the contrast between clusters can be lost. To restore this contrast, we set the target vectors to be the vectors between the initially mapped clusters in two-cone space extended by some additional magnitude. The additional magnitude a_{ij} determines how much the contrast should be enhanced and is calculated based on the distances between the clusters before and after the initial projection to the viewer's space. We compare the distances o_{ij} between the source clusters in LUV space to the distances between mapped clusters in a modified target space. The function $F()$ modifies the target space colors in two-cone space by projecting them to their simulated LMS colors on the viewer's surface and then converting them to LUV space. Given 2D coordinates in two-cone space, we use Brettel et al. [Bret 97] to simulate what the color deficient viewer sees, giving us the corresponding 3D LMS colors on the viewer's surface. The conversion of mapped clusters from LMS space to LUV space makes it easy to compare contrasts between the mapped clusters to contrasts between the source clusters by comparing distances in LUV space. For the sigmoidal weight ψ_{ij}, we center our sigmoid on $c = 15$, determined empirically from our image data.

Critical pairs determine which pairs of clusters are favored over other pairs for stronger contrast enhancement during the optimization. These pairs are given higher weights in the optimization to match their targets and lower weights to match the initial mapping. Critical pairs are those that lose significant color contrast in the eyes of a color deficient viewer. To find critical pairs, we find pairs that overlap more in the target space, or two-cone space, than they do in the source space, or LUV space. Since the target and source spaces are different, we compare fractional overlap measures as described in Section 4.4.

The achromaticity term E_L ensures that achromatic source colors are kept neutral. The matrix L in this term projects colors onto the subspace perpendicular to the neutral axis, or the $\begin{bmatrix} 1 & 1 \end{bmatrix}^{\mathrm{T}}$ axis in two-cone space.

For achromatic colors, this means that their distances to the neutral axis should be the same before and after the optimization. To make sure only achromatic colors are held to this soft constraint, achromaticity weights are defined for each row as a Gaussian function of chroma values of the LUV source colors, with $\sigma = 10$. The achromaticity term is described in more detail in Section 4.4.

We solve the least squares optimization in Equation 7.1 for the optimal cluster colors on the color deficient surface that preserve original color contrasts and maintain naturalness. We constrain the optimal clusters to be within the target space bounds, which are the $[0, 1]$ bounds of the two-cone space. The optimal cluster means $\mathbf{x}_i$ are each constrained to be within $[r_{m_i}, 1 - r_{m_i}]$ where r_{m_i} is the radius of the mapped cluster in two-cone space.

7.2.5 Blending and Final Projection

The blending step transfers the optimization results to the pixels to produce an output image in two-cone space. The optimization gives us new cluster locations in two-cone space, and we compare these optimal locations to the initial mapped cluster means in two-cone space to get vector translations for each cluster. Instead of applying these vector translations to each cluster's constituent pixels, each pixel is translated by a weighted blend of cluster movements, as described in Section 4.5. This updates each pixel's initially mapped color in two-cone space. The updated pixels are then clipped to the $[0, 1]$ bounds of the two-cone space.

7.2.6 Display

Our method solves for an output image in the target space, which is the two-cone space of the color deficient viewer. To simulate how the viewer would see our output image, we use Brettel et al. [Bret 97] to convert the output image from two-cone space to LMS colors on the viewer's surface. Then, to view the output image on a typical display, we convert the LMS output image to sRGB. The conversion from LMS cone space to sRGB space transforms LMS values into XYZ space and then into linear RGB space followed by a nonlinear gamma correction. This nonlinear transformation accounts for the display gamma of a standard sRGB display device.

7.3 Results

Our results show that our method is able to preserve contrasts from the original image that were lost when viewed by a particular color deficient viewer. The strong color contrast between the orange flower and green leaves in Figure 7.4 is greatly reduced in the eyes of a protanope, as shown in the simulated image. Our output shows an enhanced contrast between the flower and the leaves using only colors from the viewer's surface. Figure 7.4 also contains an Ishihara test pattern, a dot pattern with specially colored dots used for testing color deficiencies. In this pattern, people with red-green color deficiencies, protanopes and deuteranopes, are not able to see the number "6" that people with normal color vision can see. Our method enhances contrast so that the "6" is visible to a deuteranope.

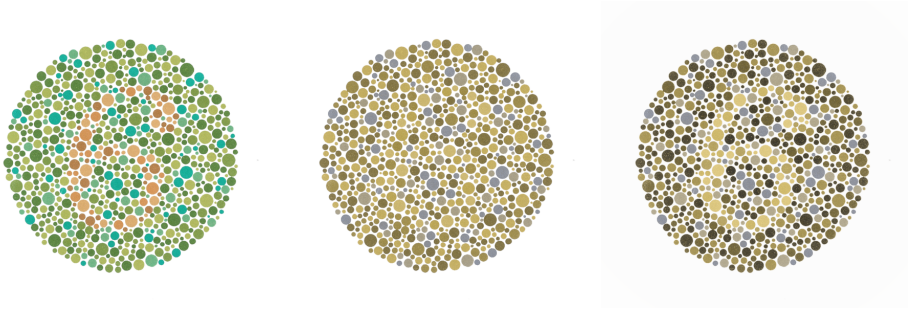
Our method aims to portray original contrasts while maintaining naturalness by staying close to the simulated colors of a particular viewer. Figure 7.5 shows how some green and blue jelly beans would look to a tritanope. The green and blue jelly beans that are easy for normal viewers to pick out are hard to distinguish for a tritanope. The method of Rasche et al. [Rasc 05a] changes the green jelly beans to orange and the blue ones to gray. In our output, the contrast between the jelly beans is enhanced slightly without deviating too far from the colors the tritanope naturally sees. Also in Figure 7.5, the red flowers on the green background lose their contrast to a protanope. While Kuhn et al. [Kuhn 08a] recolor the flowers to blue, our method enhances the contrast slightly while staying closer to the colors the protanope naturally sees.

7.4 Summary

Recoloring images for color deficient viewers is a problem that fits within our framework. The information loss comes from a projection of 3D colors onto the surface of distinguishable colors for a dichromatic viewer. To a dichromat, or one who only has two normally functioning cones out of three, some colors can appear very similar though they are very different for a viewer with normal vision. In the extreme case, this information loss leads to metamerism. When optimizing images for color deficient viewers, the goal is to preserve contrasts that viewers with normal vision can see easily. We apply our general method to optimizing images for dichromats such that original contrasts are preserved while remaining faithful to what the dichromat sees initially. Our results enhance visible contrast for the color

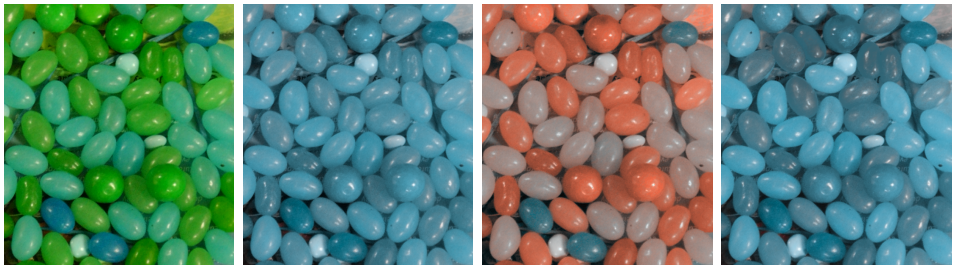
deficient viewer while staying close to the colors the viewer naturally sees.

(a) original (b) protanope (c) our output



(d) original (e) deuteranope (f) our output

Figure 7.4: **Preserving contrast.** The strong color contrast between the orange flower and green leaves in (a) is not seen by a protanope, simulated in (b). Our output (c) preserves the contrast between the flower and leaves. When the Ishihara test pattern in (d) is viewed by a deuteranope, simulated in (e), the number "6" is not discernable. In our output in (f) a deuteranope is able to see the "6". Our method preserves contrast within the viewer's space of distinguishable colors. *Original images courtesy of (a) Rasche et al. [Rasc 05b] and (d) Wikipedia [Ishi].*



(a) original (b) tritanope (c) Rasche et al. [Rasc 05a] (d) our output

(e) original (f) protanope (g) Kuhn et al. [Kuhn 08a] (h) our output

Figure 7.5: **Maintaining naturalness.** The green and blue jelly beans in (a) look very similar to a tritanope, simulated in (b). Rasche et al. [Rasc 05a] change colors of the jelly beans to orange and gray in (c). Our method enhances contrast slightly while staying close to the colors the tritanope naturally sees. Our output in is (d). For a protanope, the red flowers in (e) look similar to the background, as simulated in (f). Kuhn et al. [Kuhn 08a] drastically change the flower color to enhance contrast with the background in (g). Our method makes the flowers distinguishable from the background while staying close to the colors the protanope naturally sees. Our output is in (h). *Original images courtesy of (a) Rasche et al. [Rasc 05b] and (e) Kuhn et al [Kuhn 08b].*

Chapter 8

Multispectral and Multiprimary Image Fusion

8.1 Introduction

A multispectral image consists of n spectral bands where each band contains information from a narrow range of wavelengths in the electromagnetic spectrum. A multiprimary image is similar to a multispectral image in that it also contains n images, one for each primary of a device. Conventional RGB cameras have three primaries, but a multiprimary image with more than three channels can be constructed with such a camera by capturing the scene multiple times, each time with a different filter in front of the camera to shift the primaries. The primaries of a multiprimary image can have wider transmission functions than the narrower ranges of wavelengths typical of bands in multispectral images. More generally, multispectral and multiprimary images are multichannel images comprising n images of a scene. Other multichannel images could consist of a tristimulus image augmented with additional channels of information as long as there is pixel correspondence between all channels. A depth map is an example of such an additional channel. When $n > 3$, the numerous channels of multichannel images make it difficult to visualize on standard displays.

Multispectral image fusion makes it possible to visualize an immense amount of spectral data as a single fused image. For any multichannel image with n bands, there are n different images and n values for each pixel. It is hard for humans to mentally integrate n images and extract relevant information [Wils 97]. For example, Figure 8.1 shows the six bands of a multiprimary image which are hard to visualize as they are laid out. It is easier to visualize the scene in a single image, but to do this, we must reduce the n-channeled image to a tristimulus image viewable on a standard display. Figure 8.2 shows different combinations of channels used to put together a tristimulus image. Each of the n images can contain different information, and we might not see all the information we want in a tristimulus image.

Therefore, many existing algorithms have been developed to reduce the dimensionality of multispectral data to a lower dimension image containing task-relevant information.

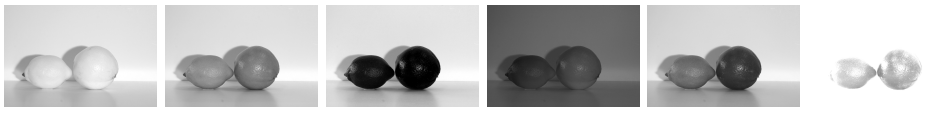


Figure 8.1: **Six channels of a multiprimary image.** The first three channels are from the standard RGB image. The last three channels are from the shifted primary image captured with a blue filter in front of the camera.

Figure 8.2: **RGB channel replacement.** Different tristimulus images are created from the multiprimary data in Figure 8.1 by replacing each of the RGB channels with one of the six bands from the multiprimary image. The first two images are the original captures without and with the blue filter. The other three images are different combinations of the six bands as follows: (3,4,5), (1,5,4), and (4,5,3). These false color representations manage to capture the contrast between the real and fake pieces of fruit, but none of them have the natural coloring of the original capture, which, unfortunately, does not show contrast between the metameric fruit.

In an n dimensional multispectral, multiprimary, or multichannel image with $n > 3$, there is more information than can be represented in a tristimulus image, and some information will be lost due to the dimensionality reduction. The standard procedure to convert multispectral data to RGB values consists of multiplying the spectral responses by color matching functions and integrating over the visible wavelengths. The schematic in Figure 8.3 shows how the input spectral response $S(\lambda)$ is multiplied by each color matching function and then integrated to get each R, G, and B value. This standard integration procedure can lead to metamerism, a phenomenon in which different spectral responses map to the same tristimulus values. Also during this process, the spectral bands outside the visible range

are neglected, so spectral responses with the same visible response but different responses outside the visible range will also exhibit metamerism and map to the same tristimulus values. This means that details outside the visible range are lost. A similar procedure happens in the human visual system, resulting in our inability to distinguish between metamers and see beyond the visible range. In our multiprimary example, placing a filter in front of the camera in order to capture the scene with a second set of primaries essentially modulates the transmission functions of the original primaries. The data captured under shifted primaries can highlight differences between materials that were metamers under the original primaries. When observing the natural image captured with the original primaries, these differences are not evident. In general, for a multichannel image with $n > 3$, all dimensions cannot be fully represented in a tristimulus image. Inevitably, transformations to a tristimulus space will lead to metamerism and a loss of information.

Multispectral, multiprimary, and multichannel image fusion fit within our framework for problems involving information loss. Since n dimensions are reduced to m, this is the most general case in the framework. When the purpose is visualization on a standard three-primary display, we need to output a tristimulus image. This means $m = 3$, and the target space is sRGB space. In multispectral and multiprimary image fusion, the loss of information due to the dimensionality reduction can lead to metamerism, where different spectral responses are represented by the same tristimulus values. Similarly, for multichannel image fusion, the exclusion of additional dimensions of information in the output tristimulus image can lead to metamerism. This information loss and occurrence of metamerism is common among all the applications in the framework.

We apply the method outlined in Chapter 4 to multispectral and multiprimary image fusion. Given multispectral or multichannel data, we use this method to create a tristimulus image for visualization on a standard display. Our goal is to output an image that represents local contrasts from all spectral bands, or input bands, without deviating too far from what looks natural to a human observer. First, we want to distinguish between metamers by enhancing contrast between areas with different spectral responses whose contrast is not easily seen in the natural image. Second, we want to do so while remaining close to the natural image, or natural mapping, obtained by the standard integration so that the output still looks like a plausible, realistic version of the scene. For example, we would like to preserve the difference between the metameric fruit in the multiprimary image in Figure 8.2. Here, the false color representations certainly depict

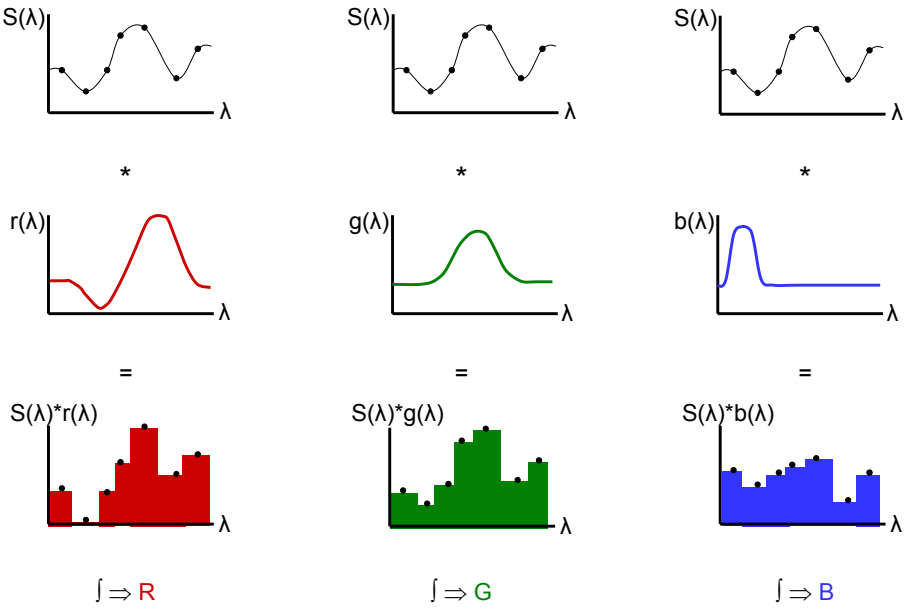
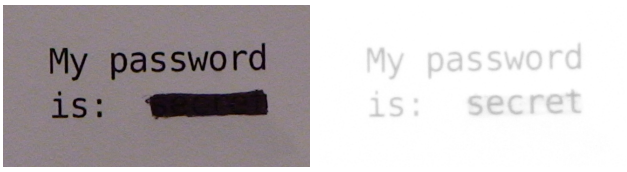


Figure 8.3: **Spectral to RGB projection.** This schematic illustrates the standard procedure for calculating RGB tristimulus values from a spectral response. The spectral response $S(\lambda)$ is multiplied by each of the RGB color matching functions $r(\lambda)$, $g(\lambda)$, $b(\lambda)$, and the results are integrated to get the R, G, and B tristimulus values.

the difference between the fruit, but the images are exceedingly different from the natural image of the scene and do not look realistic. The goal of our method is to preserve the contrast between the metameric fruit without straying unecessarily far from their natural coloring. Figure 8.4 shows an example of a visible and near-infrared image pair. Our goal is to fuse this image pair such that the details from the near-infrared image and the colors from the visible image are simultaneously preserved. These two goals to preserve local contrasts and stay close to a natural mapping are the goals of the framework.

Our results show how we are able to create tristimulus images from multispectral, multiprimary, or multichannel data that preserve contrasts between metamers while staying close to the natural representations that a



(a) visible RGB (b) near-infrared

Figure 8.4: **Visible and near-infrared image pair.** The goal is to fuse these images into an image that looks similar to the visible image but contains the details from the near-infrared image.

human observer would see. We fuse visible and near-infrared images of the same scene to generate an output that looks similar to the visible image but contains details from the near-infrared channel. As a multichannel image fusion example, an RGB and depth image pair is combined. We also apply our method to the multiprimary image in Figure 8.1 and multispectral images with 31+ spectral bands. Similar to the results of the other applications in our framework, our results show contrast enhancement between metamers while remaining close to the standard mapping to the tristimulus space.

8.2 Method

We apply the general method described in Chapter 4 to multispectral, multiprimary, and multichannel image fusion. In this section, we explain the method in terms of multichannel image fusion and include application-specific details. First, we cluster together pixels that are similar in spectral response and spatial coordinates. Then, for the clusters, we solve for new tristimulus colors that preserve contrasts from the n channels. Finally, we transfer the new colors back to the pixels in the blending step.

8.2.1 Projection to Target Space

To obtain a representation of what the input spectral data looks like to a human observer, we project the multispectral or multiprimary image to a tristimulus RGB image following the standard calculations. This is our initial projection from source space $\mathcal{S}$ to target space $\mathcal{T}$, where $\mathcal{S}$ is the spectral space of the input image in $[0, 1]$ in $\mathbb{R}^n$ and $\mathcal{T}$ is sRGB space in $[0, 1]$ in $\mathbb{R}^3$. Note that the target space is the nonlinear sRGB space rather than

the linear RGB space, and we will use "RGB", in this chapter, to refer to the nonlinear sRGB space. When the input data contains R, G, and B channels, we simply take those channels and discard the rest. For pairs of images such as visible RGB and near-infrared (NIR) images, the input data has the structure (R, G, B, NIR), so our initial projection simply leaves the NIR channel out since it is outside the visible range. The same is done for RGB and depth image pairs; the depth channel is omitted. For a multiprimary image with channels containing the original RGB primaries of the device, the original capture is kept, and the shifted primary data is left out. In each of these cases, the RGB image representing the scene as a human would see it is the initial standard mapping. We project spectral data to sRGB by multiplying the spectral response by the CIE 1931 2° XYZ color matching functions and integrating to get XYZ tristimulus responses which are then converted to linear RGB and then to sRGB. If the multispectral image represents scene reflectances, such as the Metacow image [Fair] or those from the Columbia Multispectral Image Database [Yasu 08], then we multiply the spectral responses by the illuminant and the XYZ color matching functions before integrating. In our examples, we use a D65 illuminant. Projecting spectral information to the sRGB tristimulus space in this manner yields a natural image of the scene as seen by a human viewer. This standard mapping tells us where local contrasts in the spectral data have been lost for the viewer. Our optimization aims to preserve these contrasts while staying close to the natural colors of the initial mapping.

8.2.2 Clustering

In order to separate the image into areas that might exhibit local contrast, we cluster the pixels in spatio-chromatic space $\mathcal{U}$ in $\mathbb{R}^{n+2}$. Each pixel's feature vector is $\mathbf{u} = (\mathbf{s}, x', y')$, consisting of its spectral vector $\mathbf{s}$ in $[0, 1]$ augmented by its weighted spatial coordinates (x', y'). For an input visible and near-infrared image pair, $\mathbf{s}$ is (R, G, B, NIR) with each component in $[0, 1]$. Similarly, for an input RGB and depth map image pair, $\mathbf{s}$ is (R, G, B, depth) with each component in $[0, 1]$. Details on the clustering, including the weight on the spatial dimensions that balances the tradeoff between spectral space and spatial space are in Section 4.2. Once the clusters are established, we calculate statistics to define ellipsoids for the clusters. We calculate means, covariance matrices, and approximate radii for three sets of clusters: the clusters in spatio-chromatic space $\mathcal{U}$, the clusters in source space $\mathcal{S}$ in $\mathbb{R}^n$, and the mapped clusters in the RGB target space $\mathcal{T}$ in $\mathbb{R}^3$. The means, covariance matrices, and radii for the mapped RGB clusters are

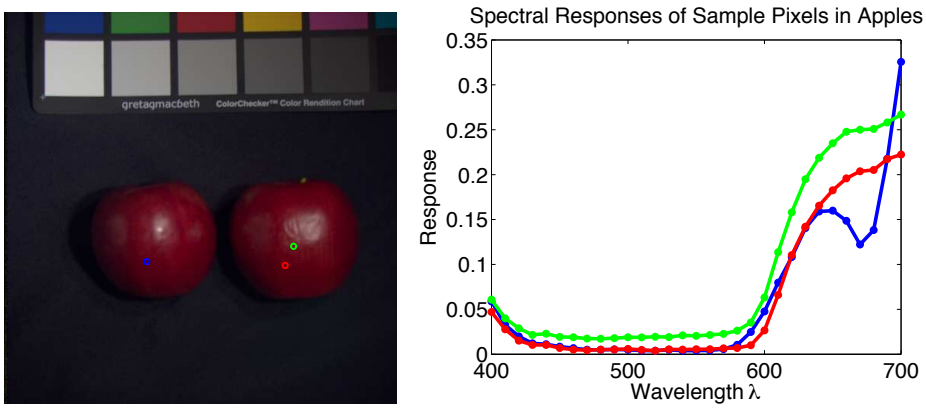
calculated from the data in the mapped RGB image.

When the number of spectral bands, n, becomes large, it becomes more difficult to measure small differences in spectral response shape caused by changes within a single channel. For example, spectral responses from the real and fake apples in Figure 8.5(a) are plotted in blue and red in Figure 8.5(b). The spectral responses are very similar yet significantly different around 670 nm where the blue response has a slight dip followed by a sharp spike at 700 nm. The spectral response for a nearby pixel on the second apple is plotted in green. When compared, the two responses (red and green) from the same apple are more similar in shape than the responses (red and blue) for the real and fake apples. Despite this similarity, the Euclidean distance between the responses from the second apple is larger than the distance between the responses from the two different apples. This is because small per-channel differences between the second apple's pixels accumulate to be greater than the distance between the real and fake apples' responses, despite the small dip and sharp spike. As n gets larger, the contrast in a single channel has less impact on the distance metric, and vertical shifts in spectral responses, corresponding to changes in brightness, have more impact. In this example, we would like our clustering algorithm, which forms clusters based on Euclidean distance, to separate the two apples into different clusters. Our goal is to separate pixels into clusters based on their relative differences in spectral responses instead of based on their differences in brightness. To do this, we normalize the spectral data when $n > 4$ so that each spectral response has a length of 1. In these cases, the source space $\mathcal{S}$ becomes the normalized spectral space, and normalized spectral responses are used throughout the rest of the method in ellipsoid calculation, optimization, and blending.

Some of our example images are of laboratory setups with metameric objects. For these examples, we want to specifically influence contrast between the objects in the scene. Since our goal is to distinguish between a few objects in a simple setup rather than between local features present in complex, natural images, we are less interested in clustering by local region. Therefore, for these simple setups, we reduce the spatial contribution in $\mathcal{U}$ by multiplying the spatial coordinates (x', y') by 10^{-6}.

8.2.3 Graph Creation

We create the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of clusters, and the edges in $\mathcal{E}$ connect spatially close clusters. The edges are determined by the dilation-and-neighbor-counting procedure described in Section 4.3.



(a) sample pixels (b) spectral responses

Figure 8.5: **Why normalize spectral responses?** The spectral responses from sample pixels on the real and fake apples in (a) are plotted in (b). The red and green responses from the apple on the right are more similar in shape than the the red and blue responses from the two different apples. In comparison, the blue response has a dip at 670 nm and a sharp spike at 700 nm. Despite these differences, the Euclidean distance between the red and blue responses is actually larger than the distance between the red and green responses. Since we cluster pixels together based on Euclidean distance, we normalize the spectral responses to have unit length so that the red and green responses are clustered together while the red and blue responses are not. This results in separate clusters for each apple. *Original image from the Columbia Multispectral Image Database courtesy of Yasuma et al. [Yasu 08].*

For images of laboratory setups, the task is to distinguish between the objects in the scene. Since these simple setups contain objects that might not be spatially touching, we consider more spatially distant pairs by adding edges to the two-neighbors from the original graph.

8.2.4 Optimization

During the optimization, we solve for new RGB colors for the clusters that best preserve local contrasts from all bands and natural colors from the initial mapping to RGB space. (Note that, as mentioned before, the target RGB space is the nonlinear sRGB space and not the linear RGB space.) We

solve the linear least squares problem from Equations 4.1, 4.2, 4.5, and 4.8, repeated here and simplified, for the output RGB colors $\mathbf{x}$ that minimize

$$\begin{aligned} \mathbf{x} &= \arg\min_{\mathbf{x}} E_T + \lambda_M E_M + \lambda_L E_L \\ E_T &= \sum_{(i,j)\in\mathcal{E}} \tau_{ij}((\mathbf{x}_j - \mathbf{x}_i) - \mathbf{t}_{ij})^2 \\ E_M &= \sum_{i\in\mathcal{V}} \tau_i(\mathbf{x}_i - \mathbf{m}_i)^2 \\ E_L &= \left\| L(\mathbf{x} - \mathbf{m}) \right\|^2 \end{aligned} \tag{8.1}$$

where E_T is the term that preserves contrast, E_M is the term that keeps colors close to the initial natural colors, E_L is the term that preserves the neutrality of achromatic colors, and λ_M and λ_L are parameter weights. The hue term and previous iteration term from Equation 4.1 have been excluded for this application as linear planes in RGB space do not correspond to constant hue planes.

The target vectors $\mathbf{t}_{ij}$ are designed to enhance contrast between clusters and are defined in Equation 4.3, which is repeated below.

$$\begin{aligned} \mathbf{t}_{ij} &= \frac{m_{ij} + a_{ij}}{m_{ij}} \mathbf{m}_{ij} \\ a_{ij} &= k \cdot S(\psi_{ij}(o_{ij} - \left\| F(\mathbf{m}_j) - F(\mathbf{m}_i) \right\|)) \\ \psi_{ij} &= \frac{1}{1 + e^{-\kappa(\left\| F(\mathbf{m}_j) - F(\mathbf{m}_i) \right\| - c)}} \\ S(x_{ij}) &= \frac{\max(0, x_{ij})}{\max_{(i,j)\in\mathcal{E}} x_{ij}} \end{aligned} \tag{8.2}$$

To enhance contrasts, we would like the initial vectors between RGB clusters to be extended in magnitude. Therefore, we set the target vectors to the initial mapped vectors $\mathbf{m}_{ij}$ lengthened by additional magnitude a_{ij}. The additional magnitude amount a_{ij} depends on how far apart the mapped clusters are in RGB space compared to how far apart the original clusters are in spectral space. We set $F()$ to the identity function (i.e. $F(\mathbf{m}) = \mathbf{m}$), so the amount a_{ij} is based on the discrepancy between spectral distances and RGB distances. The sigmoid is a function of distances in RGB space in $[0, 1]$. We center the sigmoid on $c = 0.15$, which we determined empirically from our image data.

Critical pairs are cluster pairs that need the most contrast enhancement. For these pairs, the initially mapped clusters in RGB space overlap more

than the original clusters in spectral space. Critical pairs are found by comparing fractional overlap amounts in RGB space to fractional overlap amounts in spectral space, as detailed in Section 4.4.

The achromaticity term E_L ensures that neutral colors remain neutral. More specifically, we preserve the achromaticities of the initially mapped RGB clusters, the achromaticities that the human visual system would see. Therefore, mapped RGB colors that are close to the gray axis get high weights while the others are assigned near-zero weights. The matrix L in this term projects colors onto the plane perpendicular to the $\begin{bmatrix}1 & 1 & 1\end{bmatrix}^{\mathrm{T}}$ gray axis in RGB space.

We solve the least squares optimization in Equation 8.1 for cluster colors within the target space, which is the RGB cube. Since the RGB cube bounds are $[0, 1]$ in each dimension, we constrain each cluster's output color $\mathbf{x}_i$ to be within $[r_{m_i}, 1 - r_{m_i}]$, where r_{m_i} is the radius of the mapped cluster in RGB space.

8.2.5 Blending and Final Projection

The optimization gives us new RGB cluster means from which we can calculate cluster movements from the initially mapped cluster means. The optimization results are transferred back to the pixels by shifting each pixel in the initially mapped RGB image by a weighted blend of cluster movements, according to Section 4.5. To make sure the pixels are within the RGB target space, they are clipped to $[0, 1]$.

8.2.6 Display

Since the target space is the sRGB color space, the output image is ready for display on an sRGB monitor. The nonlinear sRGB space includes a gamma curve with an overall gamma of 2.2. No further processing on the output image is necessary to account for a display gamma of 2.2.

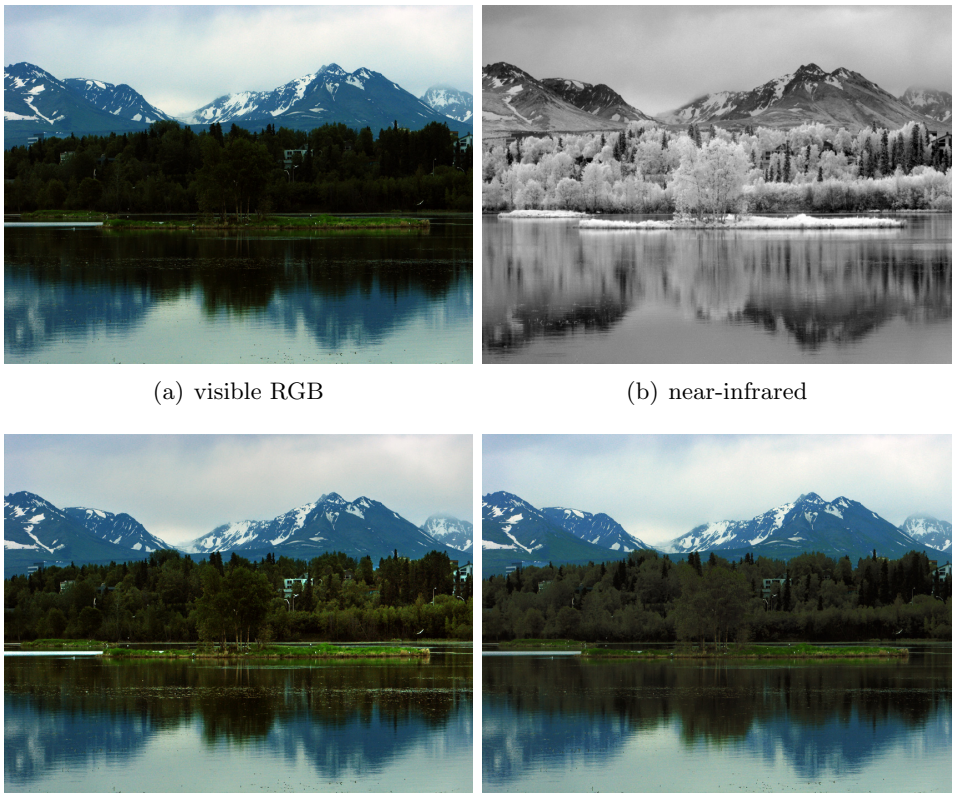
8.3 Results

Compared to the standard transformation of a multispectral, multiprimary, or multichannel image with $n > 4$ to a tristimulus image, our results show that our method is able to preserve details from the discarded channels and distinguish between spectral metamers while staying close to the natural projection. Figures 8.6 and 8.7 show our results for combining visible RGB and near-infrared pairs. The standard projection discards the near-infrared

information since this information is not perceived by the human visual system. Near-infrared light reflects strongly off foliage, as it does in Figure 8.6, giving us more detail in the trees. Since near-infrared light has a longer wavelength than the visible wavelengths, it penetrates through the atmosphere farther than visible light because of Rayleigh scattering. Also, near-infrared light penetrates some inks, as it does in Figure 8.7(h) to reveal the password that was hidden under black marker. In these examples, we are able to capture the details in the trees, the contours of the hidden mountain, and the hidden password. When combining an RGB image with a depth map in Figure 8.8, contrast in the RGB image is enhanced according to depth. The contrast between the tree stump and the background is enhanced, and the colors of the tree stump change to show the gradation of the depth map. The contrasts from the depth map, which are lost during the truncation to RGB, are incorporated into the RGB image without straying too far from the natural projection. Figure 8.9 shows our results for multiprimary image fusion of the six-primary example from Figure 8.1. This test scene contains a plastic lemon and a real orange. The six channels of this image come from two images of the scene: the original RGB image, which serves as the natural projection, and the image captured with a blue filter in front of the camera. The lemon and orange have similar RGB colors but different spectral responses as evident from their contrast in the blue-filtered image. Our method reduces the six-primary image to an RGB image where the fruits are different colors from each other but still similar enough to their natural colors. Similarly, in Figure 8.10, the real and artificial foods have different spectral responses, but this is not easily seen by the human visual system or preserved when spectral data is converted to RGB. In the standard projection of the spectral data to RGB (first column), the real and fake foods have similar colors. Despite the similar tristimulus appearance, the mean spectral responses for the areas in the blue and red crop boxes show that the spectral responses for the real and fake foods are in fact different, especially in the 650-700 nm range. The real and fake foods are spectral metamers, and our method enhances the contrast between them to represent their difference in spectral response. In the Metacow example in Figure 8.11, the two halves of each cow are metamers under illuminant D65. Our method is able to distinguish between these spectral metamers by enhancing the contrast between the cow halves. For multispectral and multiprimary image fusion, our method is able to preserve contrasts from the source space that are not visible in the standard projection to RGB while staying close to the standard projection, maintaining naturalness.

8.4 Summary

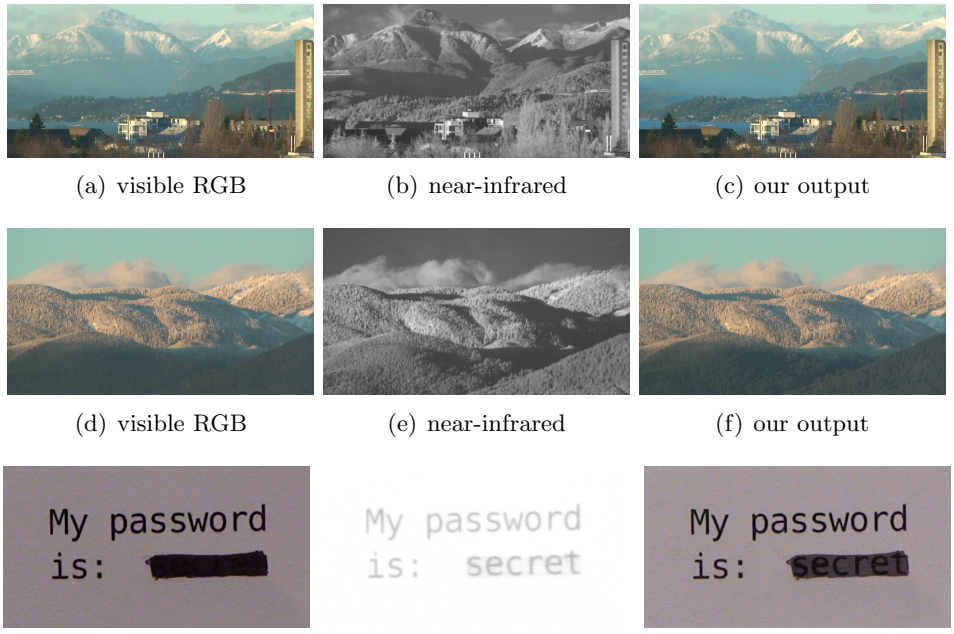
Multispectral or multiprimary image fusion fits within the context of our framework. The input image has n channels, where n is larger than three, and the goal is to reduce these n dimensions to the three dimensions of a tristimulus image viewable on a standard display. From the dimensionality reduction, there is an information loss, which leads to metamerism. The standard projection of spectral data to RGB space yields a tristimulus image which is the natural represention of how the scene would look to a human observer. We apply our method to multispectral image fusion with the goal of creating a tristimulus image that preserves details in the unseen near-infrared channels and distinguishes between metamers while staying close to the natural colors from the standard projection. Similar to the other applications in our framework, our goal is to preserve local contrasts from the source space within the constraints of the target space while staying close to the natural mapping for a natural result.



(a) visible RGB (b) near-infrared

(c) Zhang et al. [Zhan 08] (d) our output

Figure 8.6: **Visible RGB + near-infrared fusion.** The near-infrared image (b) contains more details in the trees than the visible RGB image (a) since near-infrared light reflects strongly off foliage. Our method combines the visible RGB image with the near-infrared image resulting in our output (d), which shows more details in the dark trees while staying close to the natural colors of the visible image. Zhang et al.'s [Zhan 08] output (c) also improves contrast in the trees, but their method is designed only for visible/near-infrared fusion. *Original image courtesy of Zhang et al. [Zhan 08].*

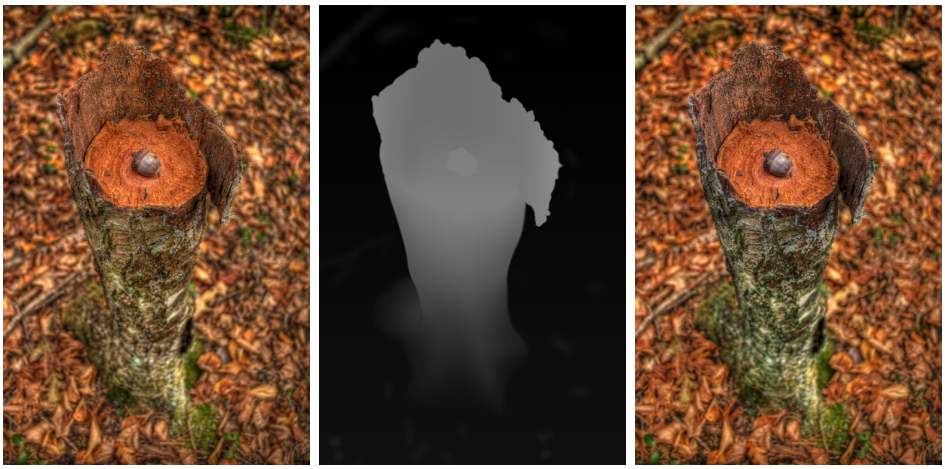


(a) visible RGB (b) near-infrared (c) our output

(d) visible RGB (e) near-infrared (f) our output

(g) visible RGB (h) near-infrared (i) our output

Figure 8.7: **Visible RGB + near-infrared fusion.** These examples combine visible RGB and near-infrared images. Near-infrared light penetrates farther through the atmosphere than visible light in accordance with Rayleigh scattering. It also penetrates through some inks as it does in (h), revealing the password under the black marker. Our output images show the contours of the hidden mountain in (c), additional details in the trees in (c) and (f), and the hidden password in (i).

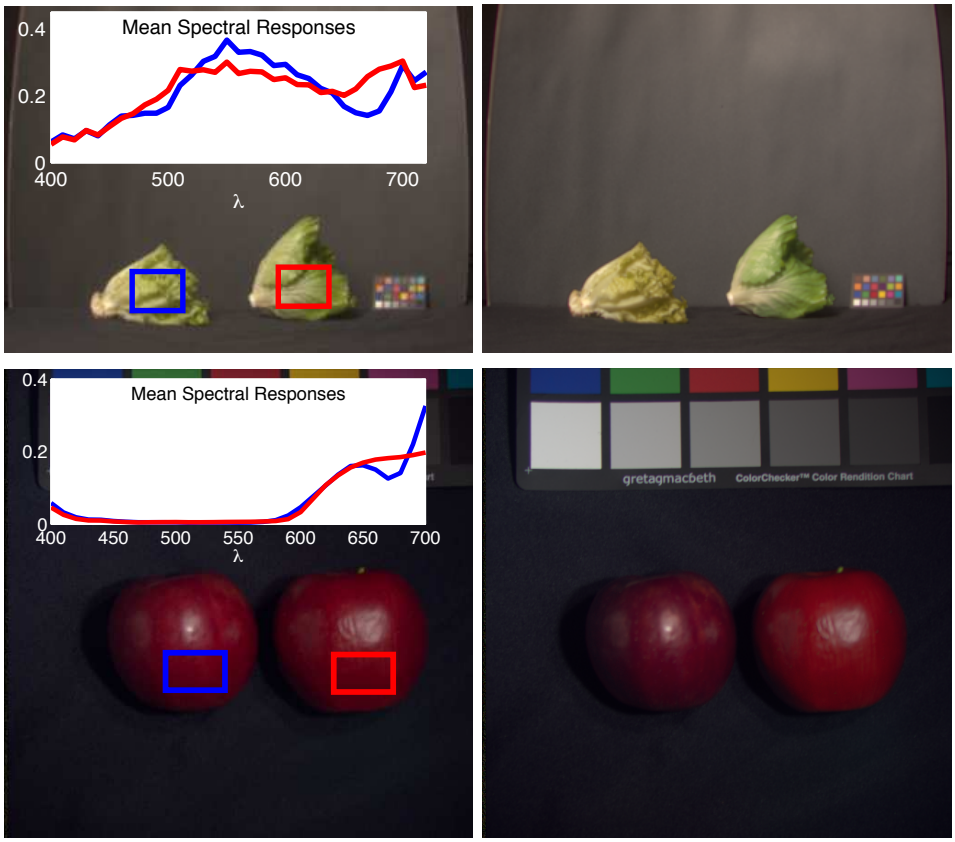


(a) RGB (b) depth image (c) our output

Figure 8.8: **RGB + depth fusion.** We combine an RGB image (a) with a depth image (b) in this multichannel image fusion. In comparison to the original RGB image, our output (c) shows an enhanced contrast between the stump and the background, corresponding to the sharp contrast in the depth map. The colors of the tree stump are also modified, following the gradation in the depth map. *Original image courtesy of Justin Manteuffel.*

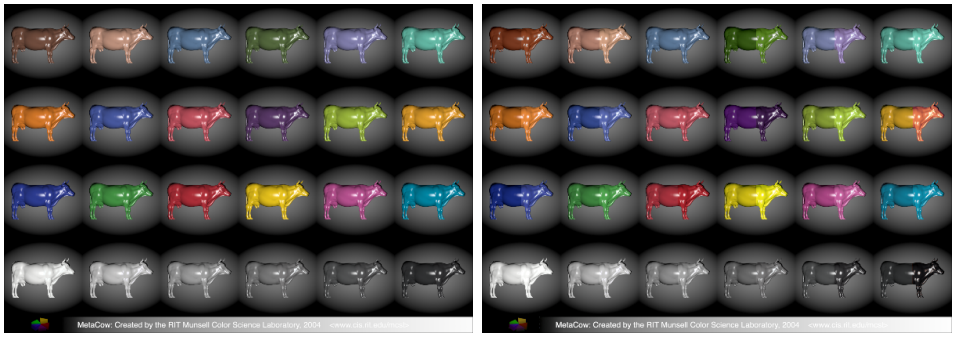
(a) original RGB, no filter (b) blue filter (c) our output

Figure 8.9: **Multiprimary image fusion.** This multiprimary image has six channels. The first three channels are from the original RGB image (a) captured under the primaries of a conventional RGB camera. The last three channels are from the shifted-primary image (b) captured with a blue filter in front of the same camera. The blue filter serves to shift the primaries of the camera. The artificial lemon and real orange have similar colors in the original RGB image but different colors in the blue-filtered image making them metamers in RGB. Our result (c) captures this difference by enhancing the contrast between the metameric fruit while keeping close to the natural colors of the original RGB image.



(a) spectral converted to RGB (b) our output

Figure 8.10: **Multispectral image fusion.** Our method enhances contrast between the metameric foods in these examples. The multispectral images of real and fake foods contain 33 bands in 400-720 nm for image of lettuce and 31 bands in 400-700 nm for image of apples. The first column shows the result of the standard projection of spectral data to RGB. The mean normalized spectral response is calculated for each crop window (blue and red) and plotted at the top of each image. For the lettuce, the blue spectral response has a dip around 670 nm and a slightly steeper peak at 550 nm in comparison to the red response. For the apples, the blue response also dips around 670 nm and then increases steeply. In both cases, the blue and red spectral responses differ, yet the RGB colors look similar, making them metamers. Our method enhances contrast between them to distinguish between metamers while staying close to the natural colors, producing the results in the second column. *Original image (row 2) from the Columbia Multispectral Image Database courtesy of Yasuma et al. [Yasu 08].*



(a) spectral converted to RGB (b) our output

Figure 8.11: **Multispectral image fusion.** Our input is 39 spectral bands of Metacow image data in 380-760 nm in 10 nm increments. We render the spectral image under a D65 illuminant and project the spectral data to RGB in (a). The two halves of each cow are metamers, and under D65, the RGB colors for the two halves look the same. The underlying spectral data differs, and our method distinguishes between the metameric halves by enhancing the contrast between them, yielding our result (b). *Original image courtesy of the RIT Munsell Color Science Laboratory [Fair].*

Chapter 9

Limitations and Parameters

9.1 Limitations

Despite our method's general applicability to problems in the framework, it is not without limitations. These limitations include the determination of the target vector direction in the optimization, the spatial neighbor creation in the graph creation, and the dependence on the parameter specifying the number of clusters to use.

9.1.1 Determination of Target Vector Direction

The most serious limitation with our method occurs in the optimization step in the determination of the target vector direction. During the optimization step, the target vectors $\mathbf{t}_{ij}$ are set to the initial mapped vectors $\mathbf{m}_{ij}$ lengthened by some additional magnitude a_{ij}. For the applications that use Equation 4.3 (partially repeated below) to define target vectors, the direction of the target vectors is set to the direction of the initial mapped vectors.

$$\mathbf{t}_{ij} = \frac{m_{ij} + a_{ij}}{m_{ij}} \mathbf{m}_{ij} \tag{9.1}$$

These mapped vectors are the vectors between the initially mapped cluster means in the target space; these clusters have already been projected to the target space. In the case of isoluminant colors mapping to the same luminance value, the mapped vector between them will theoretically be the zero vector. This means that our optimization will not be able to enhance the contrast between them since their target vector will be the zero vector. In practice, however, exact numerical equivalence is unlikely, and the contrast between the isoluminant colors will be enhanced according to the additional magnitude, but the direction of the enhanced contrast will be arbitrarily determined by numerical noise.

Figure 9.1 shows an example of when the target vector directions, as determined by our initial mapped vector directions, fail to give us the result we desire: the enhancement of contrast between the green and purple ele-

ments in the image. The purple cluster, when mapped to grayscale, is both lighter than some of the green clusters and darker than other green clusters. This causes the target vectors to be in both directions, where some targets aim to make the purple elements lighter and some targets aim to make the purple elements darker. The result is that the purple elements are not made distinguishable from the green elements.

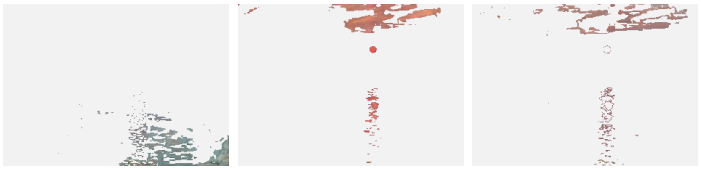
(a) original (b) luma (c) our output (d) consistent vector directions

Figure 9.1: **Mixed target vector directions for green and purple elements.** The contrast between the purple and green elements in (a) is lost in the luma conversion (b). Since the purple cluster is both lighter and darker than the green elements and the optimization cannot choose whether the purple flowers should be made lighter or darker than the background, our result (c) fails to distinguish between the color elements. If some target vector directions were reversed, indicating that the purple cluster should be lighter than the green clusters, then we would get the output in (d).

Inconsistent target vector directions for similarly colored cluster pairs also causes unwanted contrast changes for the image in Figure 9.2. In this example, the red clusters are generally lighter than the blue clusters. However, cluster 10, which is blue, poses an exception to this consensus by having a lighter luma value than its red neighbors, clusters 4 and 8, as shown in Table 9.1. This means the target vector directions for pairs (4,10) and (8,10) aim to make the blue cluster lighter than the red clusters. Fixing the inconsistency by manually switching the directions of these vectors results in a better output image.

One possible solution to target vector direction inconsistency is to parameterize the chromatic space and assign vector directions according to the parameterization, as in the method of Gooch et al. [Gooc 05]. Gooch et al.'s color to gray method only needs to determine the sign of target vector in the 1D grayscale space. They do this by parameterizing the color difference space shown in Figure 9.3(b), based on a parameter θ. The sign of each

(a) original (b) luma (c) our output (d) consistent vector directions



(e) cluster 10 (f) cluster 4 (g) cluster 8

Figure 9.2: **Inconsistent target vector directions.** Generally, the red clusters in (a) are slightly lighter than the blue clusters in terms of luma (b). However, cluster 10 (e) differs from this trend as it is a blue cluster that has lighter luma values than its red neighbors, clusters 4 (f) and 8 (g). This leads to cluster 10 having a lighter output color than the red clusters, causing the water in the bottom right part of the image (c) to be lighter than desired. When fixing the target vector inconsistencies, the output (d) has better contrast between the reflection and the water. *Original image courtesy of Gooch et al. [Gooc 05].*

target vector, determined by a dot product rule, is based on the orientation of vectors between colors in the a^*b^* plane shown in Figure 9.3(a). Since the parameterization is only based on the a^*b^* chromaticities, the target vector signs are independent of the lightness values and, therefore, independent of the standard mapping to grayscale. This means that following this parameterization could lead to large luminance reversals. Furthermore, this specific parameterization and dot product rule only applies to 1D target vectors and does not work for multidimensional target spaces.

Another possible solution and avenue for future work is to determine the target vector direction based on the image data. Perhaps there is a trend in the image data that we can follow. For example, the image in Figure 9.2(a) contains red and blue colors, and the general trend is for the red colors to be lighter than the blue colors. If we can extract this rule, then we can set our target vector signs to make the red elements lighter than the blue elements.

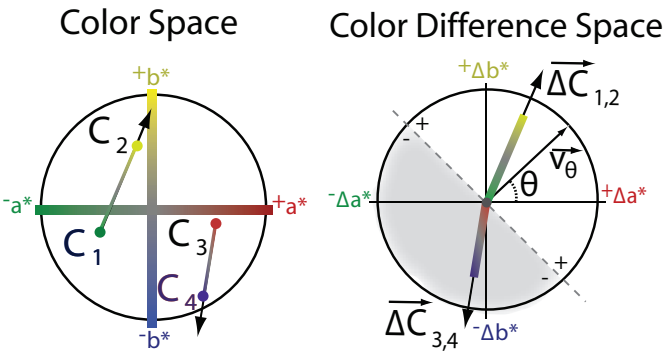
Table 9.1: **RGB and mapped cluster means.** The blue cluster (10) has a slightly higher luma value than the red clusters (4 and 8). Since our target vector directions are assigned based on the mapped vector directions, this is the source of the undesired modification of cluster 10 in our output in Figure 9.2(c).

Cluster	RGB Means	Mapped Means (luma)	Luma Values in [0,1]
10			0.5364
4			0.5337
8			0.5274

For a 3D color target space, we could examine the existing distribution of vectors between colors in the mapped image. To determine the target vector direction between metameric colors, we could either try to conform to the existing data to maintain naturalness or create vectors that are orthogonal to the data to create the most contrast. Extracting the general trends from image data or determining how to use the existing data are problems to be solved.

9.1.2 Spatial Neighbor Creation

Sometimes an image might contain visually important features, and it is desirable to preserve the contrast between these features. Another limitation of our method is that the contrast between important regions that are relatively far apart will not be preserved in our optimization. This happens because the contrast between these regions is not represented in our graph. To avoid extraneous global relationships, our graph only considers the local contrast between regions that are spatially close and does not contain edges between more distant regions. Unfortunately, this means the contrast between salient yet distant regions will not be preserved. For example, the two heads of lettuce in Figure 9.4 are not spatially close enough for their contrast to be represented in the graph. We would like the clusters representing green parts of the left and right heads of lettuce to form cluster pairs in the graph.



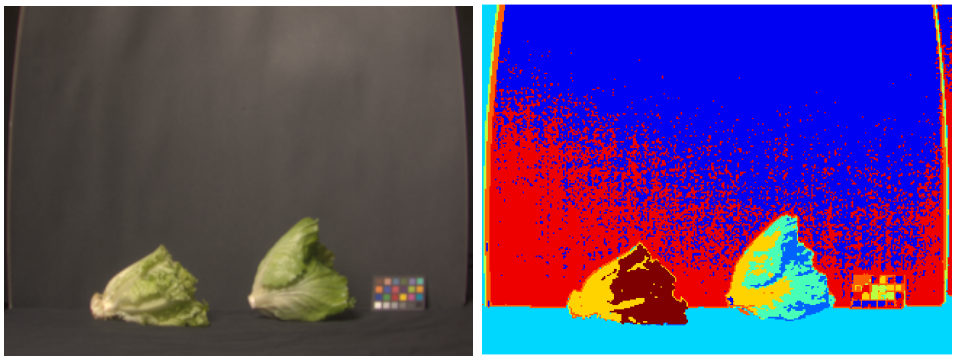
(a) color space (b) color difference space

Figure 9.3: **Color space parameterization used by Gooch et al. [Gooc 05].** Gooch et al. use vectors between clusters in the a^*b^* chromatic plane of CIE LAB space to determine their target vector directions. For color pairs (C_1, C_2) and (C_3, C_4) in the a^*b^* plane in (a), their chromatic difference vectors $\overrightarrow{\Delta C_{1,2}}$ and $\overrightarrow{\Delta C_{3,4}}$ are shown in the color difference space in (b). The color difference space is parameterized by the parameter θ, which determines the orientation of the vector $\overrightarrow{v_\theta}$. The sign of each target vector is set to the sign of $(\overrightarrow{\Delta C_{i,j}} \cdot \overrightarrow{v_\theta})$. *Images courtesy of Gooch et al. [Gooc 05].*

One solution to this problem is to add edges connecting the two-neighbors or three-neighbors of the graph. Since the heads of lettuce share common neighbors (both heads spatially touch the background cluster), they will be connected in the graph after the two-neighbors are added. Adding edges to more distant neighbors in the graph introduces more global information into our optimization. Therefore, we only add the two-neighbors for scenes in which we would like to preserve important relationships between more distant regions. Alternatively, we could try to find the visually salient regions in an image and only add edges between these significant regions to the graph. It could be possible to classify these regions based on statistics such as size, density, color, or spatial location.

9.1.3 Dependence on Number of Clusters

The number of clusters to create during the clustering step is a parameter of our method. The output of our method is dependent on the number of clusters specified by the user. We choose the number of clusters for each



(a) RGB image (b) cluster image

Figure 9.4: **Non-spatial neighbors.** The two heads of lettuce are the salient features in this multispectral image converted to RGB (a). The cluster image (b) shows that there are different clusters for the green parts of each lettuce head. Even though these clusters do not spatially touch, they are visually important, and we would like their contrast to be represented in our graph. Our graph creation step will not create edges between distant pairs like the two lettuce heads, but we can add extra edges to the graph connecting clusters to their second neighbors (the neighbors of their neighbors).

image so that the resulting clusters generally resemble image features but also so that their means are representative of their constituent pixels. If there are not enough clusters, then multiple features might be combined into the same cluster. This is the case in Figure 9.5 where the blue and purple matches are grouped into the same cluster. The RGB cluster mean is not very representative of either match head color, which means contrast measurements using this cluster mean will be inaccurate. In the output, the contrast between the blue and purple matches is not preserved because they are treated as a single entity. If there are too many clusters, then relationships between important image features could be lost. For example, the red and green peppers in Figure 9.6 are key objects in the scene, and it is desirable to preserve the chromatic contrast between them. With too many clusters, the peppers are split into multiple clusters, and the main relationship between the peppers is not represented. When there are too many clusters, the local contrasts between clusters approach pixel level contrasts instead of representing the relationships between areas or regions. It

is important to choose the number of clusters so that the resulting clusters represent the image features whose contrast between each other we wish to preserve. One possible solution would be to use a cluster splitting rule. During clustering, if a cluster is too large (i.e. represents too many colors and/or spatial regions), it should be split. With this rule, the number of clusters parameter could start out small, and the clusters would split as needed. Another possible solution would be to set up a grid structure in the source space. For many applications the source space is a color space such as LUV, and each grid cell would mark an area of similar colors. Then, we could count the number of grid cells that contain more than a threshold number of pixels and use this as an initial number of clusters. Cluster mean initialization could also come from the information in each grid cell. Cluster splitting could be used to determine the final number of clusters.

9.2 Parameters

Parameters in the optimization step, described in Section 4.4, control the relative weighting of the optimization terms and control target vector lengths. The linear least squares optimization problem from Equation 4.1 is repeated here.

$$\mathbf{x} = \arg\min_{\mathbf{x}} E_T + \lambda_M E_M + \lambda_H E_H + \lambda_L E_L + \lambda_P E_P. \tag{9.2}$$

The parameter weights on the optimization terms are λ_M, λ_H, λ_L, and λ_P, which correspond to weights on the regularization term, hue term, achromaticity term, and previous iteration term, respectively. The parameters k and c influence target vector lengths. A target vector's length is affected by how much additional magnitude is added to the initial mapped vector. The additional magnitude a_{ij} is calculated by Equation 4.3, which is repeated here.

$$\begin{aligned} \mathbf{t}_{ij} &= \frac{m_{ij} + a_{ij}}{m_{ij}} \mathbf{m}_{ij} \\ a_{ij} &= k \cdot S(\psi_{ij}(o_{ij} - \left\| F(\mathbf{m}_j) - F(\mathbf{m}_i) \right\|)) \\ \psi_{ij} &= \frac{1}{1 + e^{-\kappa(\left\| F(\mathbf{m}_j) - F(\mathbf{m}_i) \right\| - c)}} \\ S(x_{ij}) &= \frac{\max(0, x_{ij})}{\max_{(i,j)\in\mathcal{E}} x_{ij}} \end{aligned} \tag{9.3}$$

Additional magnitudes a_{ij} for all pairs (i, j) are weighted by a sigmoidal weight ψ_{ij} and scaled to $[0, k]$. The sigmoid for the sigmoidal weight is

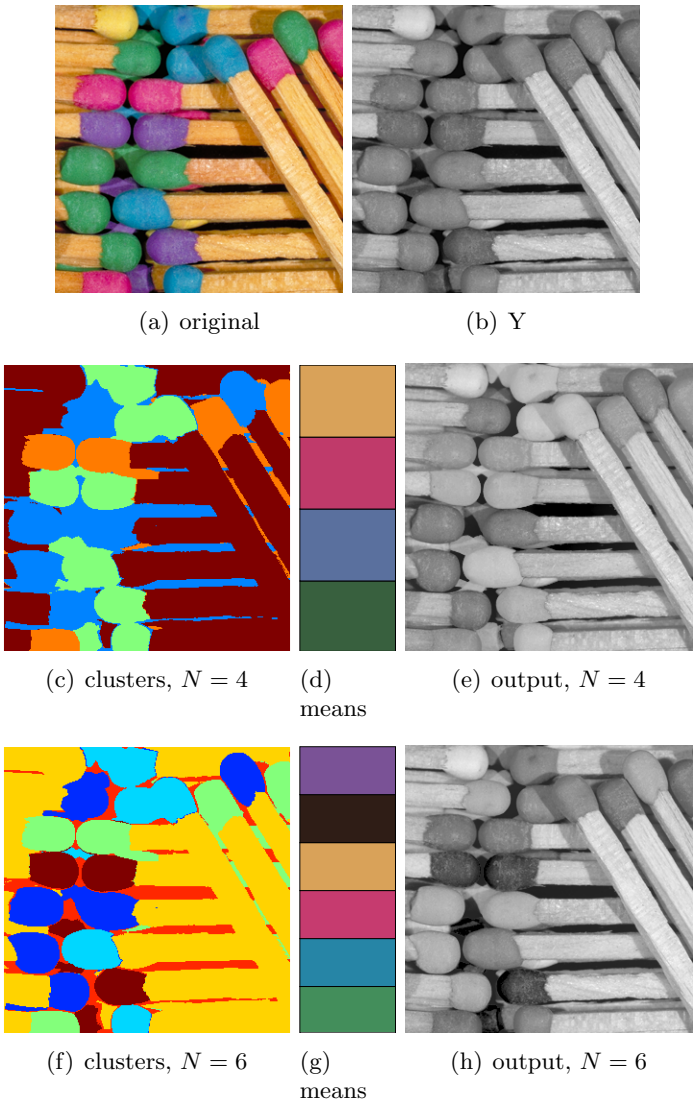
centered on c. The largest amount by which an initial mapped vector can be increased is k.

The regularization parameter λ_M and the parameter k, which specifies the maximum magnitude increase allowed, control the tradeoff between staying close to the standard mapping and deviating from it in order to enhance contrast. Figure 9.7 shows the effect of varying these parameters for the fusion of the RGB and depth image pair from Figure 8.8. As the regularization term weight λ_M increases from 0.2 to 1.0, the output image looks more similar to the standard mapping, which, in this case, is equivalent to the original RGB image. As the parameter k increases from 0.2 to 1.0, the optimization tries to match cluster differences to longer target vectors, cluster pairs are pulled farther apart, and contrast is increased in the output image. We set $\lambda_M = 0.8$ in order to stay close to the standard mapping but allow some flexibility for change. For the examples in this thesis, we choose the parameter k manually depending on how much contrast we want to add to the image. We typically choose k in the range [0.1,1.0]. For the fusion example in Figure 9.7, we let $k = 0.6$.

In order to prevent hue shifts and preserve the neutrality of achromatic colors, we set the hue term and achromaticity term weights to $\lambda_H = 4$ and $\lambda_L = 10$, which is relatively high compared to the implied unity weight on the target term. We set these weights high in order to encourage movement within the planes of constant hue rather than perpendicular to them and to strongly encourage the movement of neutral colors along the achromatic axis rather than perpendicular to it.

The sigmoidal weight in the additional magnitude calculation in Equation 9.3 comes from a sigmoid centered on $c = 0.15$ for multispectral image fusion and $c = 15$ for color to gray conversion and image optimization for color deficient viewers. The values for c are different because the sigmoid is a function of distances in the modified target space, and the modified target space is RGB space in $[0, 1]$ for multispectral image fusion, L^* (in $[0, 100]$) for color to gray conversion, and $L^*u^*v^*$ space for image optimization for color deficient viewers. The sigmoid center is determined empirically from our image data. From our images and cluster information, we manually select metameric cluster pairs whose contrast should be enhanced by our method, such as cluster pairs of real and fake food items. We also manually selected cluster pairs of items that were visibly different but had moved closer together during the initial mapping. These are pairs whose contrast we do not want to enhance because they already have sufficient contrast. Then, we plotted all of the cluster pairs from the images in a plot where the x axis is the distance between the cluster pair in the source space, and

the y axis is the distance between the cluster pair in the modified target space. Each cluster pair has a point in the plot, and the specially selected metameric cluster pairs are plotted in a different color to distinguish them from the other manually selected group of cluster pairs. From this plot we decided that a horizontal line at $y = 0.15$ (for multispectral image data) is a good choice to separate the selected metameric pairs from the other cluster pairs whose contrast was already sufficient. Many of the metameric pairs were below the line while many of the pairs with sufficient contrast were above the line. The position of this line on the y axis is the center for our sigmoid. The sigmoidal weight gives pairs that are above the line lower weights than the pairs that are below the line. This means that pairs with sufficient contrast will not receive as much enhancement as the metameric pairs that need it more.

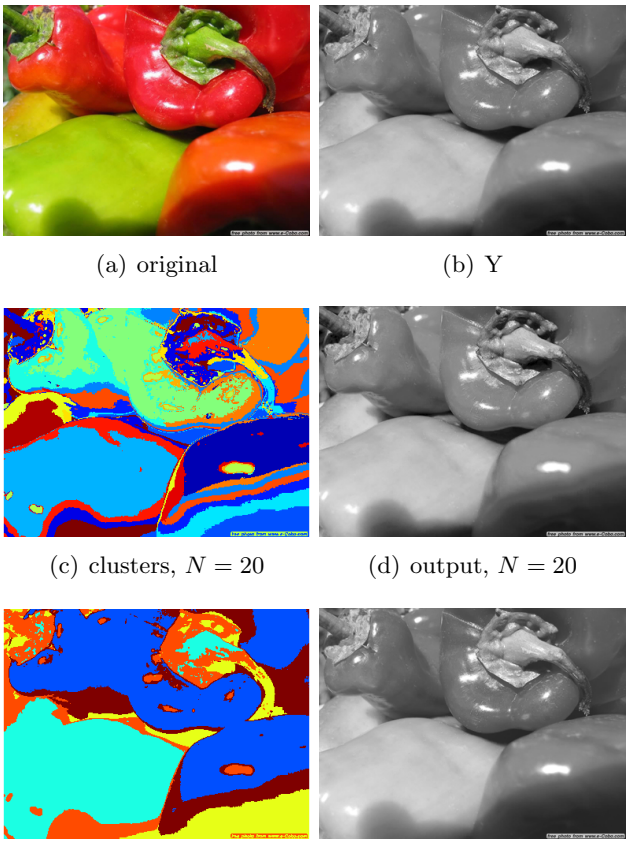


(a) original (b) Y

(c) clusters, $N = 4$ (d) means (e) output, $N = 4$

(f) clusters, $N = 6$ (g) means (h) output, $N = 6$

Figure 9.5: **Not enough clusters.** With an insufficient number of clusters, image features might be grouped together. The blue and purple matches from (a) are grouped together in a single cluster when the number of clusters N is set to 4 in (c). The cluster mean becomes a mix of the two colors and represents neither color accurately. When N is increased to 6 in (f), the blue and purple matches have their own clusters. Their cluster means better represent their underlying colors. The output (h) for $N = 6$ preserves the contrast between the blue and purple matches better than the output (e) for $N = 4$. *Original image courtesy of Grundland and Dodgson [Grun 07].*



(a) original (b) Y

(c) clusters, $N = 20$ (d) output, $N = 20$

(e) clusters, $N = 5$ (f) output, $N = 5$

Figure 9.6: **Too many clusters.** With too many clusters, it is easy for important relationships between key features in the image to not be represented. When $N = 20$ in (c), the red and green peppers are split into too many clusters. With fewer clusters in (e) where $N = 5$, the red and green peppers are each represented by a dominant cluster. The relationship between the peppers is better preserved in the output (f), where contrast is enhanced, than in the output (d). *Original image courtesy of Cadik [Cadi 08].*

Figure 9.7: **Varying parameters λ_M and k.** Here are output images for combining the RGB and depth image pair from Figure 8.8 under varying parameters. The regularization term weight λ_M increases in each row from top to bottom, making the output to stay closer to the standard mapping. The parameter k increases in each column from left to right, increasing contrast in the output. Both parameters are set to the values $\{0.2, 0.4, 0.6, 0.8, 1.0\}$. *Original RGB and depth image pair courtesy of Justin Manteuffel.*

Chapter 10

Future Work

Our cluster-based method has potential applications in other areas of color image processing besides the four transformation problems discussed within this thesis. An obvious direction of future work would be to apply our method to other problems that fit within the framework. Tone mapping, for example, is one of these problems. Alternatively, it would be interesting to extend our cluster-based method to apply to other related problems such as color transfer or expansion problems. Expansion problems include colorization, gamut expansion, dynamic range expansion.

Our framework was developed to emphasize the relationship between these four problems: color to gray, color gamut mapping, image optimization for color deficient viewers, and multispectral image fusion. Our hope is that the connection between these problems inspires others to develop general methods that apply to any of the problems in the framework or to adapt existing single-problem methods to work for other problems that fall within the framework.

10.1 Other Problems Within Framework

One obvious direction of future work is to apply our method to other problems within the framework. One such problem is tone mapping. High dynamic range (HDR) imaging offers a way to capture a scene that contains details in a larger luminance range than can be captured within the range of a traditional camera sensor. High dynamic range images typically also have a larger range than can be displayed entirely on a conventional display. To prepare the high dynamic range image for display on a lower dynamic range device, tone mapping is performed to reduce the dynamic range of the image data.

Given a high dynamic range image, the goal of tone mapping is to compress the image into a lower dynamic range while retaining the details from the high dynamic range data. One way to capture a high dynamic range image is to take a sequence of images with varying exposures and combine them. The multiexposure sequence shown in Figure 10.1 represents a

dataset with a high dynamic range. In the longest exposure on the left, the details inside the church are visible while the window areas are saturated. In the short exposure on the right, details in the windows are visible while the church interior is extremely dark. The goal of tone mapping is to compress the details from the high dynamic range image into a single lower dynamic range image which can then be visualized on a conventional display device. This compression problem involves a loss of information due to the dynamic range reduction; in this respect, it is similar to the other transformation problems in this thesis and fits nicely within our framework. For our example sequence, we would like the lower dynamic range output to contain details from both the church interior and the windows. To do this, we could use our method to preserve local contrasts from the high dynamic range source space within the constraints of the lower dynamic range target space. Since preserving local contrasts is a goal of our framework, it would be natural to apply our method to tone mapping. The second goal of our framework could be used to keep the result close to the result of a global tone mapping operator, for example. Since the reduction in dynamic range corresponds to information loss due to different volume constraints on the source and target spaces and not due to dimensionality reduction, our cluster-based method could be used to preserve both the relationships between the clusters as well as intra-cluster detail.

10.2 Related Problems

We could potentially apply an adaptation of our method to related problems such as color transfer. The goal of color transfer algorithms is to recolor a given image using the colors from another input image. For each input image, i.e. the image to recolor and the image with the color palette, we could create our cluster graph that represents the colors in each image plus the local contrasts between neighboring regions. Then, the next step would be to associate clusters from one image with clusters from the other image, indicating which colors we want to transfer to each cluster. This step is still left to be determined as future work. During the optimization we could solve for new cluster colors that preserve the local contrasts in the source image to be recolored while staying close to the color palette of the target image. Using our optimization would help preserve the image structure of the source image by maintaining local contrasts while recoloring the clusters with the colors of the target image. While the optimization would help preserve relationships between regions, the cluster-based nature of our method would

Figure 10.1: **Multiple exposures of a high dynamic range scene.** The images in this sequence vary in exposure by two stops and represent a high dynamic range dataset. In the longest exposure on the left, the details inside the church are visible while the window areas are saturated. In the short exposure on the right, details in the windows are visible while the church interior is extremely dark. The goal of tone mapping is to produce a low dynamic range image in which both the details from the church interior and the windows can be visualized in a single image on a conventional display device. *Images courtesy of Debevec and Malik [Debe 97].*

help preserve intra-cluster detail.

10.3 Expansion Problems

As future work, it would be interesting to see if we could extend our method to expansion problems such as colorization, gamut expansion, and dynamic range expansion.

Colorization. Given a grayscale image, colorization is the problem of adding color to the grayscale image. Levin et al. [Levi 04] use user scribbles to colorize a grayscale image. This is similar to the color transfer problem discussed in Section 10.2. In this case, we would like to transfer the colors of a color image to a grayscale image. Given input in the form of either user scribbles or a color image, perhaps we could extend our cluster-based method to apply to colorization in a manner similar to the adaptation suggested for color transfer.

Gamut Expansion. New technology for wider gamut displays is emerging in the form of multiprimary displays. Sharp has recently introduced a

commercially available four-primary display called the Aquos Quattron. In addition to the traditional red, green, and blue primaries in an LCD display, Sharp has added a fourth primary for yellow. Even more recently, Sharp has created a five-primary display called QuintPixel [Tomi 10] which adds a cyan primary to the previous four. These additional primaries give these displays a wider color gamut than that of a traditional three-primary display. They are able to reproduce more colors than a conventional display including the yellow of a sunflower and an emerald green sea color, which are out-of-gamut colors for a conventional display. With the advent of these new displays, the question of how to create content for these wider gamut displays arises. Also, how could we expand existing content from a smaller sRGB gamut to a wider color gamut? This is the problem of gamut expansion, which could be another potential application if we extended our cluster-based approach. For gamut expansion, we would like to expand data that exists in a smaller source space to a larger target space. Perhaps a cluster-based approach would be appropriate in this case in order to maintain intra-cluster detail and the relationships between clusters.

Dynamic Range Expansion. High dynamic range displays have a larger dynamic range than conventional displays. To adapt existing content for display on these devices, Rempel et al. [Remp 07] present a method for converting low dynamic range (LDR) images to high dynamic range (HDR) images. An example output of their Ldr2Hdr method is shown in Figure 10.2. Their method calculates a brightness enhancement mask which is applied independently to each color channel to expand the dynamic range monochromatically. However, this procedure does not modify the colors from the original LDR image. When this sunset scene is imaged by a conventional camera sensor, the pixels in the sun's region are saturated. From the outside of the sun to the center, each color channel saturates one at a time, and the color in the image does not represent the color from the scene. The individual color channel saturation creates quantized color rings around the sun before reaching full saturation in the white center. In this example, the R channel saturates first, clipping pixels to a face of the RGB cube. Then, the G channel saturates, clipping pixels to an edge of the RGB cube. Finally, the B channel saturates, clipping pixels to white. The Ldr2Hdr expansion does not account for this clipping and leaves the colors unmodified during the expansion. If we could apply our method to expansion problems such as gamut expansion, then perhaps with some additional information we can also apply our method to expanding the dynamic range of an image

and creating plausible color information in the saturated areas.

(a) original LDR image (b) brightness enhancement mask (c) HDR output

Figure 10.2: **Dynamic range expansion.** The method of Rempel et al. [Remp 07] expands the dynamic range of a low dynamic range (LDR) image (a) by calculating and applying the brightness enhancement mask (b) independently to each color channel, resulting in the higher dynamic range output (c). Two virtual exposures of the HDR image are displayed in a split-screen manner in (c) to convey the dynamic range of the image. *Images courtesy of Rempel et al. [Remp 07].*

Chapter 11

Conclusion

In conclusion, we present a unified framework to solve a wide variety of color space transformation problems using the same method. We apply our method to the specific problems of color to gray conversion, color gamut mapping, image optimization for color deficient viewers, and multispectral and multiprimary image fusion. These transformation problems have some common characteristics; they all require a transformation of the input image from its source color space to a target color space. Also, each transformation problem has its own straightforward, standard mapping to the target space, but it often incurs a loss of information which can lead to metamerism. Our general method applies to each of these transformation problems.

Our method maps an image from a source color space to a target color space in a way that preserves the local contrasts from the source space while staying faithful to the standard mapping. Our results for each of the applications show that our method preserves local contrast while maintaining the target appearance defined by the standard mapping. To do this, we solve an optimization that matches target differences and stays close to a standard projection to the target space. Our method is semi-local because it operates on clusters, modifying local contrasts between clusters. This helps avoid halo artifacts and gradient artifacts that could arise in pixel level methods and provides flexibility for the system to enhance local contrasts over maintaining global relationships.

As future work, our cluster-based method could be extended to applications that fit directly in the framework or to other related problems in color image processing. These applications include tone mapping, colorization, or gamut expansion. More generally, our method could be adapted to solve general mapping problems from a source space to a target space suffering from contrast loss during the standard mapping.

Bibliography

[Alsa 08] A. Alsam and M. S. Drew. "Fast Colour2Grey". In: *IS&T/SID 16th Color Imaging Conference*, pp. 342–345, 2008.

[Alsa 09a] A. Alsam and M. Drew. "Fast Multispectral2Gray". *Journal of Imaging Science and Technology*, Vol. 53, 2009.

[Alsa 09b] A. Alsam and I. Farup. "Colour Gamut Mapping as a Constrained Variational Problem". *16th Scandinavian Conference on Image Analysis, Lecture Notes in Computer Science*, Vol. 5575, pp. 109–118, 2009.

[Anag 07] C.-N. Anagnostopoulos, G. Tsekouras, I. Anagnostopoulos, and C.Kalloniatis. "Intelligent Modification for the Daltonization Process of Digitized Paintings". In: *Int. Conf. on Computer Vision Systems*, 2007.

[Arth 07] D. Arthur and S. Vassilvitskii. "k-means++: The Advantages of Careful Seeding". In: *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035, 2007.

[Bala 00] R. Balasubramanian, R. deQueiroz, R. Eschbach, and W. Wu. "Gamut Mapping to Preserve Spatial Luminance Variations". In: *Proc. 8th IS&T/SID Color Imaging Conf.*, pp. 122–128, 2000.

[Bala 04] R. Bala and R. Eschbach. "Spatial Color-to-Grayscale Transform Preserving Chrominance Edge Information". In: *IS&T/SID 12th Color Imaging Conference*, pp. 82–86, 2004.

[Benn 07] E. Bennett, J. Mason, and L. McMillan. "Multispectral Bilateral Video Fusion". *IEEE Trans. Image Processing*, Vol. 16, No. 5, pp. 1185–1194, 2007.

[Blac 72] H. Blackwell. "Luminance Difference Thresholds". In: *Handbook of Sensory Physiology VII/4*, pp. 78–101, Springer-Verlag, 1972.

[Bonn 06] N. Bonnier, F. Schmitt, H. Brettel, and S. Berche. "Evaluation of Spatial Gamut Mapping Algorithms". In: *Proc. 14th IS&T/SID Color Imaging Conf.*, pp. 56–61, 2006.

[Bonn 07] N. Bonnier, F. Schmitt, M. Hull, and C. Leynadier. "Spatial and Color Adaptive Gamut Mapping: A Mathematical Framework and Two New Algorithms". In: *Proc. 15th IS&T/SID Color Imaging Conf.*, pp. 267–272, 2007.

[Bonn 08a] N. Bonnier. *Contribution to Spatial Gamut Mapping Algorithms*. PhD thesis, Telecom Paristech, 2008.

[Bonn 08b] N. Bonnier, F. Schmitt, and C. Leynadier. "Improvements in Spatial and Color Adaptive Gamut Mapping Algorithms". In: *Proc. 4th European Conference on Colour in Graphics, Imaging, and Vision*, pp. 341–346, 2008.

[Bret 97] H. Brettel, F. Viénot, and J. Mollon. "Computerized Simulation of Color Appearance For Dichromats". *J. Opt. Soc. Am. A*, Vol. 14, No. 10, pp. 2647–2655, 1997.

[Burt 93] P. Burt and R. Kolczynski. "Enhanced Image Capture Through Fusion". *Proc. Fourth International Conference on Computer Vision*, pp. 173–182, 1993.

[Cadi 08] M. Cadik. "Perceptual Evaluation of Color-to-Grayscale Image Conversions". *Computer Graphics Forum (Proc. of Pacific Graphics)*, Vol. 27, No. 7, pp. 1745–1754, 2008.

[Carp 90] W. Carper, T. Lillesand, and R. Kiefer. "The use of intensity-hue-saturation transformations for merging SPOT panchromatic and multispectral image data". *Photogrammetric Engineering and Remote Sensing*, Vol. 56, pp. 459–467, 1990.

[Chav 86] P. Chavez. "Digital merging of Landsat TM and digitized NHAP data for 1:24.000 scale image mapping". *Photogramm. Eng. Remote Sens.*, Vol. 52, pp. 1637–1646, 1986.

[Chav 89] P. Chavez and A. Kwarteng. "Extracting spectral contrast in Landsat thematic mapper image data using selective principal component analysis". *Photogrammetric Engineering and Remote Sensing*, Vol. 55, pp. 339–348, 1989.

[Chip 95] L. Chipman, T. Orr, and L. Graham. "Wavelets and image fusion". *Wavelet Applications in Signal and Image Processing III*, Vol. 2569, pp. 208–219, 1995.

[Chis 99] W. Chisholm, G. Vanderheiden, and I. Jacobs. "Web Content Accessibility Guidelines 1.0". 1999. W3C Recommendation. `http://www.w3.org/TR/WCAG10/`.

[Choi 06] M. Choi. "A New Intensity-Hue-Saturation Fusion Approach to Image Fusion With a Tradeoff Parameter". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 6, pp. 1672–1682, 2006.

[CIE] "CIE Div 8 TC-03 Guidelines for the Evaluation of Gamut Mapping Algorithms Test Images". Accessed January 2010. `http://www.colour.org/tc8-03/test_images.html`.

[CIE 04] CIE. "Guidelines for the Evaluation of Gamut Mapping Algorithms". Tech. Rep. CIE 156:2004, 2004.

[Corn 70] T. Cornsweet. *Visual Perception.* Academic Press, 1970.

[Cui 09] M. Cui, A. Razdan, J. Hu, and P. Wonka. "Interactive Hyperspectral Image Visualization Using Convex Optimization". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 47, No. 6, pp. 1673–1684, 2009.

[Cui 10] M. Cui, J. Hu, A. Razdan, and P. Wonka. "Color-to-gray Conversion Using ISOMAP". *The Visual Computer: International Journal of Computer Graphics*, Vol. 26, No. 11, pp. 1349–1360, 2010.

[Das 00] S. Das and Y. Zhang. "Color Night Vision for Navigation and Surveillance". *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 1708, pp. 40–46, 2000.

[Debe 97] P. Debevec and J. Malik. "Recovering High Dynamic Range Radiance Maps from Photographs". In: *Proc. 24th Annual Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH 97)*, pp. 369–378, 1997.

[DeQu 06] R. L. DeQueiroz and K. M. Braun. "Color to Gray and Back: Color Embedding into Textured Gray Images". *IEEE Trans. on Image Processing*, Vol. 15, No. 6, pp. 1464–1470, 2006.

[Doli 09] P. Doliotis, G. Tsekouras, C.-N. Anagnostopoulos, and V. Athitsos. "Intelligent Modification of Colors in Digitized Paintings for Enhancing the Visual Perception of Color-blind Viewers". In: *IFIP Conf. on Artificial Intelligence Applications and Innovations*, 2009.

[Doug] R. Dougherty and A. Wade. "Daltonize". `http://www.vischeck.com/daltonize/`.

[Drew 09] M. S. Drew, D. Connah, G. D. Finlayson, and M. Bloj. "Improved Colour to Greyscale via Integrability Correction". In: *Proc. of SPIE-IS&T Electronic Imaging 7240*, 2009.

[Fair] M. Fairchild and G. Johnson. "METACOW: A Public-Domain, High Resolution, Fully-Digital, Noise-Free, Metameric, Extended-Dynamic-Range, Spectral Test Target for Imaging System Analysis and Simulation". `http://www.cis.rit.edu/mcsl/METACOW/`.

[Faru 07] I. Farup, C. Gatta, and A. Rizzi. "A Multiscale Framework for Spatial Gamut Mapping". *IEEE Transactions on Image Processing*, Vol. 16, No. 10, pp. 2423–2435, 2007.

[Fay 00] D. Fay, A. Waxman, M. Aguilar, D. Ireland, J. Racamato, W. Ross, W. Streilein, and M. Braun. "Fusion of Multi-Sensor Imagery for Night Vision: Color Visualization, Target Learning and Search". In: *Proc. 3rd Intl. Conf. on Information Fusion*, 2000.

[Fluc 10] D. Flück. "20 iPhone Apps for the Color Blind". 2010. `http://www.colblindor.com/2010/12/13/20-iphone-apps-for-the-color-blind/`.

[Gies 06] J. Giesen, E. Schuberth, K. Simon, D. Zeiter, and P. Zolliker. "A Framework For Image-Dependent Gamut Mapping". *Proc. SPIE*, Vol. 6058, No. 605805, 2006.

[Gooc 05] A. A. Gooch, S. C. Olsen, J. Tumblin, and B. Gooch. "Color2Gray: Salience-Preserving Color Removal". *ACM Trans. Graph.*, Vol. 24, No. 3, pp. 634–639, 2005.

[Goog] "Google Chrome Daltonize!". `https://chrome.google.com/webstore/detail/efeladnkafmoofnbagdbfaieabmejfcf`.

[Gosh 07] A. Goshtasby and S. Nikolov. "Image Fusion: Advances in the State of the Art". *Information Fusion*, Vol. 8, No. 2, pp. 114–118, 2007.

[Grun 07] M. Grundland and N. A. Dodgson. "Decolorize: Fast, contrast enhancing, color to grayscale conversion". *Pattern Recogn.*, Vol. 40, No. 11, pp. 2891–2896, 2007. `http://www.Eyemaginary.com/Portfolio/Publications.html`.

[Hari 06] H. Hariharan, A. Koschan, B. Abidi, A. Gribok, and M. Abidi. "Fusion of Visible and Infrared Images using Empirical Mode Decomposition to Improve Face Recognition". In: *IEEE International Conference on Image Processing*, pp. 2049–2052, 2006.

[Hari 96] G. Harikumar and Y. Bresler. "Feature extraction techniques for exploratory visualization of vector-valued imagery". *IEEE Transactions on Image Processing*, Vol. 5, No. 9, pp. 1324–1334, 1996.

[Harr 97] D. Harris. "Colouring sight: A study of CL fittings with colour-enhancing lenses". *Optician 5604*, pp. 38–41, 1997.

[Heck 82] P. Heckbert. "Color Image Quantization for Frame Buffer Display". *Computer Graphics*, Vol. 16, No. 3, pp. 297–307, 1982.

[Hill 05] P. Hill, D. Bul, and C. Canagarajah. "Image fusion using a new framework for complex wavelet transforms". In: *Proceedings of the IEEE International Conference on Image Processing*, pp. 1338–1341, 2005.

[Hodd 98] N. Hodd. "Putting ChromaGen to the Test". *Optometry Today*, pp. 39–42, 1998.

[Huan 07] J.-B. Huang, Y.-C. Tseng, S.-I. Wu, and S.-J. Wang. "Information Preserving Color Transformation for Protanopia and Deuteranopia". *IEEE Signal Processing Letters*, Vol. 14, No. 10, pp. 711–714, 2007.

[Huan 08] J.-B. Huang, S.-Y. Wu, and C.-S. Chen. "Enhancing Color Representation for the Color Vision Impaired". In: *Proc. ECCV Workshop on Computer Vision Applications for the Visually Impaired*, 2008.

[Ichi 03] M. Ichikawa, K. Tanaka, S. Kondo, K. Hiroshima, K. Ichikawa, S. Tanabe, and K. Fukami. "Web-Page Color Modification for Barrier-Free Color Vision with Genetic Algorithm". In: *Proc. Int. Conf. Genetic and Evolutionary Computation*, pp. 2134–2146, 2003.

[Ichi 04] M. Ichikawa, K. Tanaka, S. Kondo, K. Hiroshima, K. Ichikawa, S. Tanabe, and K. Fukami. "Preliminary Study on Color Modification for Still Images to Realize Barrier-Free Color Vision". In: *Proc. IEEE International Conference on Systems, Man, and Cybernetics*, pp. 36–41, 2004.

[Iron 05] R. Irony, D. Cohen-Or, and D. Lischinski. "Colorization by example". In: *Proceedings of Eurographics Symposium on Rendering 2005*, pp. 201–210, 2005.

[Ishi] "Ishihara Color Test". Accessed December 2010. `http://en.wikipedia.org/wiki/Ishihara_color_test`.

[Jeff 06] L. Jefferson and R. Harvey. "Accommodating Color Blind Computer Users". In: *Proc. ACM SIGACCESS Conf. Computers and Accessibility*, pp. 40–47, 2006.

[Jeff 07] L. Jefferson and R. Harvey. "An Interface to Support Color Blind Computer Users". In: *Proc. SIGCHI Conf. Human Factors in Computing Systems*, pp. 1535–1538, 2007.

[Jia 04] J. Jia, J. Sun, C.-K. Tang, and H.-Y. Shum. "Bayesian Correction of Image Intensity with Spatial Consideration". In: *In Proc. 8th European Conference on Computer Vision (ECCV)*, pp. 342–354, 2004.

[Joll 02] I. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics, 2002.

[Kim 09] Y. Kim, C. Jang, J. Demouth, and S. Lee. "Robust Color-to-gray via Nonlinear Global Mapping". *ACM Trans. Graph.*, Vol. 28, No. 5, p. 161, 2009.

[Kimm 05] R. Kimmel, D. Shaked, M. Elad, and I. Sobel. "Space-Dependent Color Gamut Mapping: A Variational Approach". *IEEE Transactions on Image Processing*, Vol. 14, No. 6, pp. 796–803, 2005.

[Koda] "Kodak PhotoCD PCD0992". Accessed January 2010. `http://r0k.us/graphics/kodak`.

[Kola 07] Ø. Kolås and I. Farup. "Efficient Hue-Preserving and Edge-Preserving Spatial Color Gamut Mapping". In: *15th Color Imaging Conference*, pp. 207–212, 2007.

[Kond 90] S. Kondo. *A Computer Simulation of Anomalous Color Vision*, pp. 145–159. Kugler & Ghedini, 1990.

[Kore 95] I. Koren, A. Laine, and F. Taylor. "Image fusion using steerable dyadic wavelet transforms". In: *In Proceedings IEEE International Conference on Image Processing*, pp. 232–235, 1995.

[Kris 09] D. Krishnan and R. Fergus. "Dark Flash Photography". *ACM Trans. Graph.*, Vol. 28, No. 3, p. 96, 2009.

[Krus 78] J. Kruskal and M. Wish. *Multidimensional Scaling*. Sage Publications, 1978.

[Kuhn 08a] G. Kuhn, M. Oliveira, and L. Fernandes. "An Efficient Naturalness-Preserving Image-Recoloring Method for Dichromats". *IEEE Trans. Visualization and Computer Graphics*, Vol. 14, No. 6, pp. 1747–1754, 2008.

[Kuhn 08b] G. Kuhn, M. Oliveira, and L. Fernandes. "An improved contrast enhancing approach for color-to-grayscale mappings". *The Visual Computer*, Vol. 24, pp. 505–514, 2008.

[Land 71] E. H. Land and J. J. McCann. "Lightness and Retinex Theory". *J. Opt. Soc. Am.*, Vol. 61, pp. 1–11, 1971.

[Lawr 10] J. Lawrence, S. Arietta, M. Kazhdan, D. Lepage, and C. O'Hagan. "A User-Assisted Approach to Visualizing Multidimensional Images". *IEEE Transactions on Visualization and Computer Graphics*, Vol. 17, No. 10, pp. 1487–1498, 2010.

[Levi 04] A. Levin, D. Lischinski, and Y. Weiss. "Colorization using Optimization". *ACM Trans. Graph.*, Vol. 23, No. 3, pp. 689–694, 2004.

[Lewi 07] J. Lewis, R. O'Callaghan, S. Nikolov, D. Bull, and N. Canagarajah. "Pixel- and region-based image fusion with complex wavelets". *Information Fusion*, Vol. 8, No. 2, pp. 119–130, 2007.

[Li 95] H. Li, B. Manjunath, and S. Mitra. "Multisensor Image Fusion Using the Wavelet Transform". *Graphical Models and Image Processing*, Vol. 57, No. 3, pp. 235–245, 1995.

[Lill 08] T. Lillesand, R. Kiefer, and J. Chipman. *Remote Sensing and Image Interpretation*. John Wiley & Sons, 2008.

[Ma 09] Y. Ma, X. Gu, and Y. Wang. "Color Discrimination Enhancement for Dichromats Using Self-Organizing Color Transformation". *Int. J. Information Sciences*, Vol. 179, No. 6, pp. 830–843, 2009.

[Mach 09] G. Machado, M. Oliveira, and L. Fernandes. "A Physiologically-based Model for Simulation of Color Vision Deficiency". *IEEE Transactions on Visualization and Computer Graphics*, Vol. 15, No. 6, pp. 1291–1298, 2009.

[Mand 96] A. Manduca. "Multispectral image visualization with nonlinear projections". *IEEE Transactions on Image Processing*, Vol. 5, No. 10, pp. 1486–1490, 1996.

[McCa 99] J. McCann. "Lessons Learned From Mondrians Applied to Real Images and Color Gamuts". In: *Proc. IS&T/SID Seventh Color Imaging Conference*, pp. 1–8, 1999.

[Megu 06] M. Meguro, C. Takahashi, and T. Koga. "Simple Color Conversion Method to Perceptible Images For Color Vision Deficiencies". In: *Proc. SPIE 6057*, pp. 432–442, 2006.

[Meye 88] G. Meyer and D. Greenberg. "Color-Defective Vision and Computer Graphics Displays". *IEEE Computer Graphics and Applications*, Vol. 8, No. 5, pp. 28–40, 1988.

[Meye 89] J. Meyer and B. Barth. "Color Gamut Matching for Hard Copy". In: *Proc. SID 89 Digest*, pp. 86–89, 1989.

[Miti 07] N. Mitianoudis and T. Stathaki. "Pixel-based and region-based image fusion schemes using ICA bases". *Information Fusion*, Vol. 8, 2007.

[Moha 08] A. Mohan, R. Raskar, and J. Tumblin. "Agile Spectrum Imaging: Programmable Wavelength Modulation for Cameras and Projectors". *Computer Graphics Forum*, Vol. 27, No. 2, pp. 709–717, 2008.

[Moro 01] J. Morovic and M. Luo. "The fundamentals of gamut mapping: A survey". *Journal of Imaging Science and Technology*, Vol. 45, No. 3, pp. 283–290, 2001.

[Moro 03] J. Morovic and Y. Wang. "A Multi-Resolution, Full-Colour Spatial Gamut Mapping Algorithm". In: *IS&T/SID 11th Color Imaging Conference*, pp. 282–287, 2003.

[Moro 08] J. Morovic. *Color Gamut Mapping*. John Wiley & Sons Ltd., 2008.

[Naka 99] S. Nakauchi, S. Hatanaka, and S. Usui. "Color gamut mapping based on a perceptual image difference measure". *Color Research and Application*, Vol. 24, No. 4, pp. 280–291, 1999.

[Neum 07] L. Neumann, M. Cadik, and A. Nemcsics. "An Efficient Perception-based Adaptive Color to Gray Transformation". In: *Proc. Computational Aesthetics*, pp. 73–80, 2007.

[Niko 01] S. Nikolov, P. Hill, D. Bull, and C. Canagarajah. "Wavelets for image fusion". In: *In Wavelets in Signal and Image Analysis, Computational Imaging and Vision Series*, pp. 213–244, 2001.

[Niko 07] S. Nikolov, E. Canga, J. Lewis, A. Loza, D. Bull, and C. Canagarajah. "Adaptive Image Fusion Using Wavelets: Algorithms and System Design". In: *NATO Security through Science Series*, pp. 243–251, 2007.

[Nune 99] J. Nunez, X. Otazu, O. Fors, A. Prades, V. Pala, and R. Arbiol. "Multiresolution-Based Image Fusion With Additive Wavelet Decomposition". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 37, No. 3, 1999.

[Petr 07] V. Petrovic and T. Cootes. "Objectively adaptive image fusion". *Information Fusion*, Vol. 8, No. 2, pp. 168–176, 2007.

[Piel 03] G. Piella. "A general framework for multiresolution image fusion: from pixels to regions". *Information Fusion*, Vol. 4, No. 4, pp. 259–280, 2003.

[Piel 09] G. Piella. "Image Fusion for Enhanced Visualization: A Variational Approach". *International Journal of Computer Vision*, Vol. 83, No. 1, pp. 1–11, 2009.

[Pire 62] M. Pirenne. "Liminal Brightness Increments". In: *The Eye, Volume 2*, pp. 159–174, Academic Press, 1962.

[Rasc 05a] K. Rasche, R. Geist, and J. Westall. "Detail Preserving Reproduction of Color Images for Monochromats and Dichromats". *IEEE Computer Graphics and Applications*, Vol. 25, No. 3, pp. 22–30, 2005.

[Rasc 05b] K. Rasche, R. Geist, and J. Westall. "Re-coloring Images for Gamuts of Lower Dimension". *Computer Graphics Forum/Eurographics*, Vol. 24, No. 3, pp. 423–432, 2005.

[Recoa] "Recommendation ITU-R BT.601-7. Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios". Accessed December 2011. `http://www.itu.int/rec/R-REC-BT.601/en`.

[Recob] "Recommendation ITU-R BT.709-5. Parameter values for the HDTV standards for production and international programme exchange". Accessed December 2011. `http://www.itu.int/rec/R-REC-BT.709/en`.

[Rein 01] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley. "Color Transfer Between Images". *IEEE Comoputer Graphics and Applications*, Vol. 21, No. 5, pp. 34–41, 2001.

[Rein 02] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda. "Photographic Tone Reproduction for Digital Images". *ACM Trans. Graph.*, Vol. 21, No. 3, pp. 267–276, 2002.

[Rein 08] E. Reinhard, E. Khan, A. Akyüz, and G. Johnson. *Color Imaging: Fundamentals and Applications*. A K Peters, Ltd., 2008.

[Remp 07] A. Rempel, M. Trentacoste, H. Seetzen, H. Young, W. Heidrich, L. Whitehead, and G. Ward. "Ldr2Hdr: On-the-fly Reverse Tone Mapping of Legacy Video and Photographs". *ACM Trans. Graph.*, Vol. 26, No. 3, p. 39, 2007.

[Rigd 99] C. Rigden. "The Eye of the Beholder - Designing for Colour-Blind Users". *British Telecommunications Engineering*, Vol. 17, pp. 2–6, 1999.

[Rock 97] O. Rockinger. "Image sequence fusion using a shift-invariant wavelet transform". In: *Proc. IEEE Intl. Conf. on Image Proc.*, pp. 288–291, 1997.

[Shet 92] V. Shettigara. "A Generalized Component Substitution Technique For Spatial Enhancement of Multispectral Images Using a Higher Resolution Data Set". *Photogrammetric Engineering and Remote Sensing*, Vol. 58, No. 5, pp. 561–567, 1992.

[Sing 08] R. Singh, M. Vatsa, and A. Noore. "Integrated Multilevel Image Fusion and Match Score Fusion of Visible and Infrared Face Images For Robust Face Recognition". *Pattern Recognition*, Vol. 41, No. 3, pp. 880–893, 2008.

[Smit 05] M. Smith and J. Heather. "A Review of Image Fusion Technology in 2005". *Proc. SPIE*, Vol. 5782, No. 29, 2005.

[Smit 06] K. Smith, G. Krawczyk, K. Myszkowski, and H. Seidel. "Beyond Tone Mapping: Enhanced Depiction of Tone Mapped HDR Images". *Computer Graphics Forum*, Vol. 25, No. 3, pp. 427–438, 2006.

[Smit 08] K. Smith, P.-E. Landes, J. Thollot, and K. Myszkowski. "Apparent Greyscale: A Simple and Fast Conversion to Perceptually Accurate Images and Video". *Computer Graphics Forum/Eurographics*, Vol. 27, No. 2, pp. 193–200, 2008.

[Soco 02] D. A. Socolinsky and L. B. Wolff. "Multispectral Image Visualization Through First-Order Fusion". *IEEE Trans. Image Process.*, Vol. 111, No. 8, pp. 923–931, 2002.

[Song 03] J. Song, S. Yang, C. Kim, J. Nam, J. Hong, and Y. Ro. "Digital Item Adaptation For Color Vision Variations". In: *Proc. SPIE 5007*, pp. 96–103, 2003.

[Suss 10] S. Süsstrunk and C. Fredembach. "Enhancing the Visible with the Invisible: Exploiting Near-Infrared to Advance Computational Photography and Computer Vision". *SID Int. Symp. Dig.*, 2010.

[Tana 10] G. Tanaka, N. Suetake, and E. Uchino. "Lightness Modification of Color Image for Protanopia and Deuteranopia". *Optical Review*, Vol. 17, No. 1, pp. 14–23, 2010.

[Toet 02] A. Toet. "Natural colour mapping for multiband nightvision imagery". *Information Fusion*, Vol. 4, No. 3, pp. 155–166, 2002.

[Toet 89] A. Toet, J. van Ruyven, and J. Valeton. "Merging thermal and visual images by contrast pyramids". *Optical Engineering*, Vol. 28, No. 7, pp. 789–792, 1989.

[Toet 90] A. Toet. "Hierarchical image fusion". *Machine Vision and Applications*, Vol. 3, No. 1, pp. 1–11, 1990.

[Toet 92] A. Toet. "Multiscale contrast enhancement with applications to image fusion". *Optical Engineering*, Vol. 31, pp. 1026–1031, 1992.

[Tomi 10] K. Tomizawa, A. Yoshida, K. Nakamura, and Y. Yoshida. "QuintPixel: Multi-Primary Color Display Systems". In: *ACM SIGGRAPH 2010 Emerging Technologies*, p. 19, 2010.

[Tren 12] M. Trentacoste, R. Mantiuk, W. Heidrich, and F. Dufrot. "Unsharp Masking, Countershading and Halos: Enhancements or Artifacts?". *Computer Graphics Forum (Eurographics)*, Vol. 31, No. 2, 2012.

[Tsag 05] V. Tsagaris and V. Anastassopoulos. "Fusion of visible and infrared imagery for night color vision". *Displays*, Vol. 26, No. 4, pp. 191–196, 2005.

[Tumb 93] J. Tumblin and H. Rushmeier. "Tone Reproduction for Realistic Images". *IEEE Computer Graphics and Applications*, Vol. 13, No. 6, pp. 42–48, 1993.

[Tyo 03] J. Tyo and R. Olsen. "Principal-components-based display strategy for spectral imagery". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 41, No. 3, pp. 708–718, 2003.

[Vien 99] F. Viénot, H. Brettel, and J. Mollon. "Digital Video Colourmaps For Checking the Legibility of Displays by Dichromats". *Color Research and Application*, Vol. 24, No. 4, pp. 243–252, 1999.

[Waki 05] K. Wakita and K. Shimamura. "SmartColor: Disambiguation Framework for the Colorblind". In: *Proc. ACM SIGACCESS Conf. Computers and Accessibility*, pp. 158–165, 2005.

[Walr 97] J. Walraven and J. Alferdinck. "Color Displays for the Color Blind". In: *Proc. IS&T/SID 5th Color Imaging Conf.*, pp. 17–22, 1997.

[Wels 02] T. Welsh, M. Ashikhmin, and K. Mueller. "Transferring color to greyscale images". *ACM Trans. Graph.*, Vol. 21, No. 3, pp. 277–280, 2002.

[Wetz 10] G. Wetzstein, W. Heidrich, and D. Luebke. "Optical Image Processing Using Light Modulation Displays". *Computer Graphics Forum*, Vol. 29, No. 6, pp. 1934–1944, 2010.

[Wils 97] T. Wilson, S. Rogers, and M. Kabrisky. "Perceptual-based image fusion for hyperspectral data". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 35, No. 4, pp. 1007–1017, 1997.

[Yang 03] S. Yang and Y. Ro. "Visual Contents Adaptation For Color Vision Deficiency". In: *Proc. IEEE Int. Conf. Image Processing*, pp. 453–456, 2003.

[Yasu 08] F. Yasuma, T. Mitsunaga, D. Iso, and S. Nayar. "Generalized Assorted Pixel Camera: Post-Capture Control of Resolution, Dynamic Range and Spectrum". Tech. Rep. CUCS-061-08, Department of Computer Science, Columbia University, 2008.

[Yock 95] D. Yocky. "Image merging and data fusion by means of the discrete two-dimensional wavelet transform". *J. Opt. Soc. Am. A*, Vol. 12, pp. 1834–1841, 1995.

[Yock 96] D. Yocky. "Multiresolution wavelet decomposition image merger of Landsat Thematic Mapper and spot panchromatic data". *Photogramm. Eng. Remote Sens.*, Vol. 62, pp. 1067–1074, 1996.

[Zhan 08] X. Zhang, T. Sim, and X. Miao. "Enhancing photographs with Near Infra-Red images". In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[Zhan 97] Z. Zhang and R. Blum. "A region-based image fusion scheme for concealed weapon detection". In: *Proceedings of the 31th Annual Conference on Information Sciences and Systems*, pp. 168–173, 1997.

[Zhao 10] Y. Zhao and Z. Tamimi. "Spectral Image Decolorization". In: *Proceedings of the 6th International Conference on Advances in Visual Computing*, pp. 747–756, 2010.

[Zoll 07] P. Zolliker and K. Simon. "Retaining Local Image Information in Gamut Mapping Algorithms". *IEEE Transactions on Image Processing*, Vol. 16, No. 3, pp. 664–672, 2007.